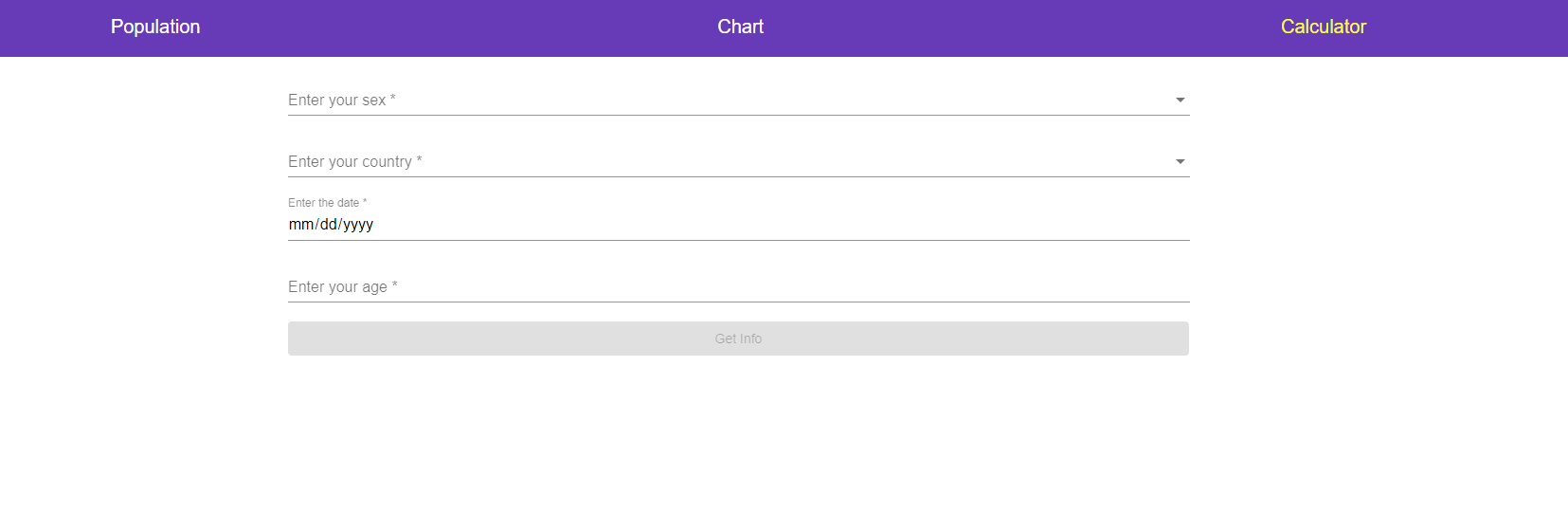
**Лабораторна робота 2\_3**

Тема: **Робота з AngularJS**

Завдання:

Реалізувати форму замовлення (повинно бути реалізовано 5 кнопок з назвою товару та ціною на кожній з них та навпроти кожної кнопки реалізовано поле для введення, в якому користувач може ввести кількість відповідного товару), використовуючи HTML, CSS і AngularJS. В окремому полі підрахувати загальну суму замовлення користувача.

Розв’язок:



Контрольні питання

1. Що таке AngularJS?

2. Назвіть основні переваги та недоліки AngularJS.

3. Назвіть основні риси AngularJS.

4. Які цілі розробки при використанні AngularJS?

**Відповіді:**

**1. AngularJS** - JavaScript-фреймворк з відкритим вихідним кодом. Призначений для розробки односторінкових додатків. Його мета - розширення браузерних додатків на основі MVC шаблону, а також спрощення тестування і розробки.

**2. Переваги AngularJS:**

* AngularJS забезпечує можливість створення односторінкового додатки в дуже чистим і обслуговує чином.
* AngularJS забезпечує можливість пов'язувати дані з HTML, таким чином, даючи користувачеві багатий і адаптивний досвід AngularJS код підходить під Unit Tests.
* AngularJS використовує впровадження залежності і використовує поділ завдань. AngularJS забезпечує повторно використовувані компоненти.
* З AngularJS, розробник пише менше коду і отримує більше функціональних можливостей.
* У AngularJS, чисті HTML-сторінки, а також контролери, написані на JavaScript робить обробку бізнес логіки.
* На додаток до всього, AngularJS додатки можуть працювати у всіх основних браузерах і смартфонах, Android і IOS.

Недоліки AngularJS:

* Чи не безпечно - будучи JavaScript фреймворком, додатки, написані в AngularJS не є безпечними. Перевірка справжності на стороні сервера і авторизація обов'язково повинні бути присутніми в додатку.
* Чи не розкладаються - Якщо користувач додатка відключає JavaScript, тоді він буде просто бачити HTML сторінку і більше нічого.

**3. Найбільш важливі основні риси AngularJS**

1. Data-binding - Це автоматична синхронізація даних між моделлю і представленням.
2. Scope - Це об'єкти, які відносяться до моделі. Вони діють як клей між контролером і видом.
3. Controller - Ці функції JavaScript, які пов'язані з конкретним Scope.
4. Services - AngularJS поставляються з декількома вбудованими служб, наприклад, $ http, щоб зробити XMLHttpRequests. Це одноелементні об'єкти, які побудовані за принципом singleton.
5. Filters - Вибрати підмножина елементів з масиву і повертає новий масив.
6. Directives - Директиви є маркерами на елементи DOM (такі як елементи, атрибути, CSS, і багато іншого). Вони можуть бути використані для створення призначених для користувача HTML тегів, які служать в якості нових, призначених для користувача віджетів. AngularJS має вбудовані директиви (ngBind, ngModel ...)
7. Templates - Це рендер виду з інформацією з контролера і моделі. Це може бути один файл (наприклад, index.html) або кілька подань на одній сторінці за допомогою «partials».
8. Routing - Це концепція перемикання видів.
9. Model View Whatever - MVC є шаблоном для поділу додатки на різні частини (так званий Model, View і Controller), кожен з різними обов'язками. AngularJS не реалізує MVC в традиційному розумінні цього слова, а скоріше щось ближче до MVVM (Model-View-ViewModel). Команда Angularjs відноситься це гумористично, як Model View Whatever.
10. Deep Linking - Глибокі посилання дозволяє кодувати стан додатки в URL, так що він може бути закладкою. Після цього додаток може бути відновлений з URL в тому ж стані.
11. Dependency Injection - AngularJS має підсистему впровадження залежності, яка допомагає розробнику легше розробляти, розуміти і тестувати.

**4. Цілі розробки:**

* Відділення DOM-маніпуляції від логіки додатки, що покращує тестованих коду.
* Ставлення до тестування як до важливої ​​частини розробки. Складність тестування безпосередньо залежить від структурованості коду.
* Поділ клієнтської і серверної сторони, що дозволяє вести розробку паралельно.
* Проведення розробника через весь шлях створення програми: від проектування призначеного для користувача інтерфейсу, через написання бізнес-логіки, до тестування.