A MINI PROJECT REPORT ON

Online car booking system

Mini project report submitted in partial fulfilment of the

Requirements for the degree of

BACHELOR OF TECHNOLOGY

In

Information Technology

Submitted by

T.Balaji (17B81A1212)

Ch.Koushik (17B81A1228)

D.Sankeerthana Reddy (17B81A1259)

Under the esteemed guidance of

Mrs.Bhagya Sri.G

Assistant Professor, IT Department

CVR College of Engineering

Department of Information Technology

CVR College of Engineering

ACCREDITED BY NBA, AICTE & Affiliated to JNTU-H

Vastunagar, Mangalpally (V), Ibrahimpatnam (M), R.R.District, PIN-501 510

2020-21

Cherabuddi Education Society's
CVR COLLEGE OF ENGINEERING
(An Autonomous Institution)
ACCREDITED BY NATIONAL BOARD OF ACCREDITATION, AICTE
(Approved by AICTE & Govt. of Telangana and Affiliated to JNT University)
Vastunagar, Mangalpalli (V), Ibrahimpatan (M), R.R. District, PIN - 501 510
Web : http://cvr.ac.in, email : info@cvr.ac.in
Ph : 08414 - 252222, 252369, Office Telefax : 252396, Principal : 252396 (O)

DEPARTMENT OF INFORMATION TECHNOLOGY

CERTIFICATE

This is to certify that the Mini Project Report entitled "Online car booking system" is a bonafide record of work carried out by T.Balaji (17B81A1212), Ch.Koushik (17B81A1228), and D.Sankeerthana (17B81A1259) under my supervision in partial fulfilment of the requirement for the degree of Bachelor of Technology in Information Technology, CVR College of Engineering.

Project Guide

Mrs. Bhagya Sri.G
Information Technology.

Head of Department

Dr. Bipin Bihari Jayasingh
Information Technology.

Project Coordinator

Mrs. Sunitha Rekha.G
Asst. Professor.
Information Technology.

ACKNOWLEDGEMENT

The satisfaction of completing this project would be incomplete without mentioning our thanks to all the people who have supported us and constant guidance and encouragement have been instrumental in its completion.

We offer our sincere gratitude to **Mrs. Bhagya Sri.G**, Asst. Professor, Department of Information Technology, CVR College of Engineering for his immense support, timely co-operation and valuable advice throughout the course of our project work.

We would like to thank the Head of Department Professor **Dr. Bipin Bihari Jayasingh**, for meticulous care and cooperation throughout the project work.

We are thankful to **Mrs. Sunitha Rekha.G**, Project Coordinator, Department of Information Technology, and CVR College of Engineering for his supportive guidelines and for having provided the necessary help for carrying forward this project without any obstacles and hindrances.

We also thank the **Project Review Committee Members** for their valuable suggestions.

DECLARATION

We hereby declare that the project report entitled “Online car booking system” submitted by us to CVR College of Engineering, in partial fulfilment of the requirement for the award of the degree of B Tech, in INFORMATION TECHNOLOGY is a record of bonafide project work carried out by us under the guidance of **Mrs.Bhagya Sri.G.**

We further declare that the work reported in this project has not been submitted, either in part or in full, for the award of any other degree or diploma in this institute or any other institute or university.

Signature of the Student

T.Balaji (17B81A1212)

Signature of the Student

Ch.Koushik (17B81A1228)

Signature of the Student

D.Sankeerthana Reddy (17B81A1259)

ABSTRACT

The main idea of the project is to make a smart online car booking system that provides customers an easy way of booking a car for a test drive slot online. This is a web application which helps a customer to book a car easily according to their choice. This project offers an effective solution where customers can view various cars those available in a showroom and select the preferred car for test-drive .When a customer is in search of a new car and goes to a showroom to buy/test drive it and finds it unavailable, it is a waste of time for him. So, instead of going to a showroom he can check the cars' availability online and reach out to the showroom if it is available and he can also book a car online.

Features included:

- Details of car available.
- Allows a customer to book a slot for test drive.
- Stores the details of a customer in a database.

TABLE OF CONTENTS

Table of Figures

1. INTRODUCTION

1.1 Literature Survey

Existing Work:

The existing system is a manual system of limitations like accuracy, expense, low speed and efficiency and unformatted outputs. In the existing system, all data processing is done manually. All the files and record books are replaced by the software system. When there are a lot of issues such as retrieval and storage of the information and keeping track of them becomes a tedious task. By implementing a computerized system, the limitation in the present system will be reduced. Manpower can be reduced to a great extent and efficiency and accuracy can be increased to manifold. More over consumption of time can be reduced to far greater extend by the implementation of the proposed system.

Advantages

- Communication
- Automation
- Money
- Time
- Information
- Efficient and Time
- Saves Money
- Better Quality of Life

Limitations of Existing Work:

● the current available system is not a fully computerized and manual system for entering

The details of a customer and accessing data and managing it.

● there is no centralized database maintenance.

● there is no easy access to records of people who are interested in buying a car.

● Administrators cannot easily navigate through the database.

● there is no specific vendor site for customers to refer to cars of various brands.

2. REQUIREMENT SPECIFICATIONS

2.1 HARDWARE REQUIREMENTS

- Laptop/Desktop
- Internet Connectivity

2.2 SOFTWARE REQUIREMENTS

- Windows/Mac OS

- Python
- Chrome/Firefox
- HTML
- CSS
- Bootstrap
- Airtable
- Flask

2.3 Functional Requirements

The project is modularized into two modules, namely,

➢ Server Module

➢ Customer Module

2.3.1 Server Module

♦The server on encountering a booking automatically saves the details of the customer in a database and sends a confirmation mail accordingly.

♦All these details of data can be accessed by an authorized personnel from the showroom.

2.3.2 Customer Module

♦A customer can book a car for test drive and also for buying.

♦All the customers should provide their details to book a car and all those details are submitted to showroom executives.

2.3 Non Functional Requirements

1. Performance: Fast response time.

2. Security: Unauthorized users should not be able to login.

3. Usability: It must be user friendly.

4. Maintainability: Easy to maintain

5. Accuracy: Must be 100% accurate.

3. DESIGN

3.1 Use-Case Diagram:

Use case diagrams display the relationship among actors and use cases of the application.

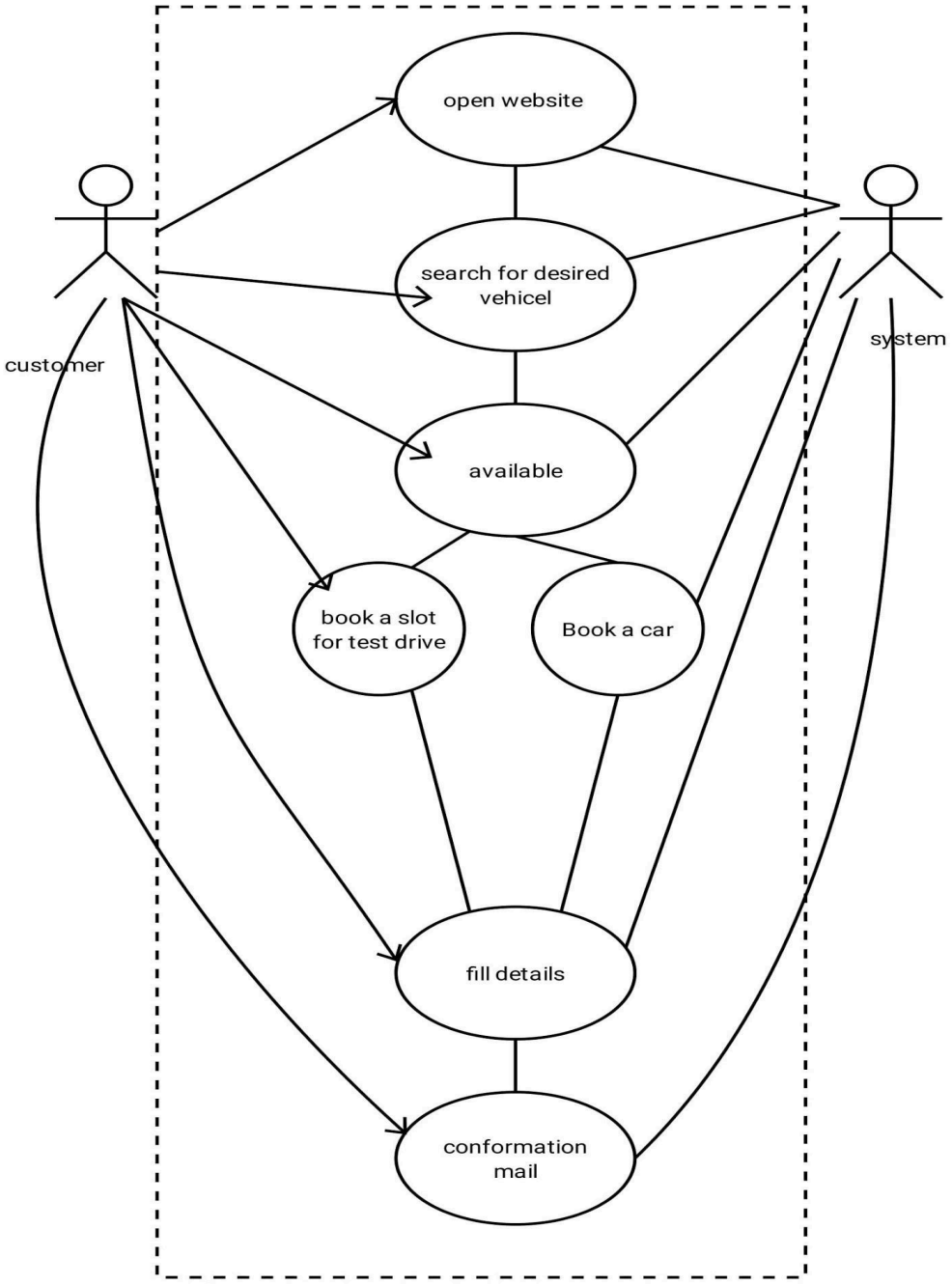


Fig 1: Use-Case Diagram of online car booking system

From the above use case there are primarily two actors:

- Customer
- System

3. Sequence Diagram:

A sequence diagram simply depicts interaction between objects in a sequential order i.e. the order in which these interactions take place. We can also use the terms event diagrams or event scenarios to refer to a sequence diagram. Sequence diagrams describe how and in what order the objects in a system function. These diagrams are widely used by businessmen and software developers to document and understand requirements for new and existing systems.

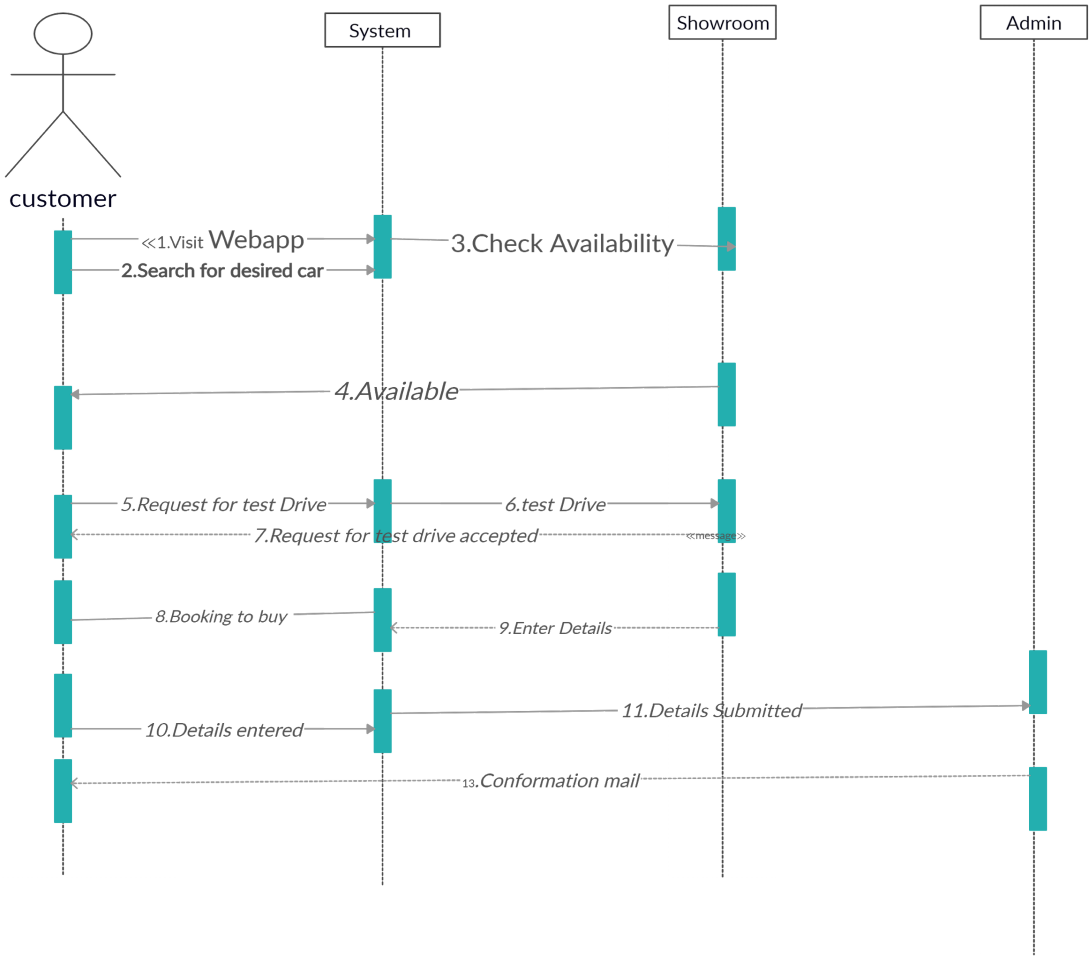


Fig 2: Sequence Diagram for online car booking system

3.3 Activity Diagram:

Activity diagram is another important diagram in UML to describe the dynamic aspects of the system. Activity-diagram is basically a flowchart to represent the flow from one activity to another activity. The activity can be described as an operation of the system. The control flow is drawn from one operation to another. This flow can be sequential, branched, or concurrent. Activity diagrams deal with all types of flow control by using different elements such as fork, join, etc.

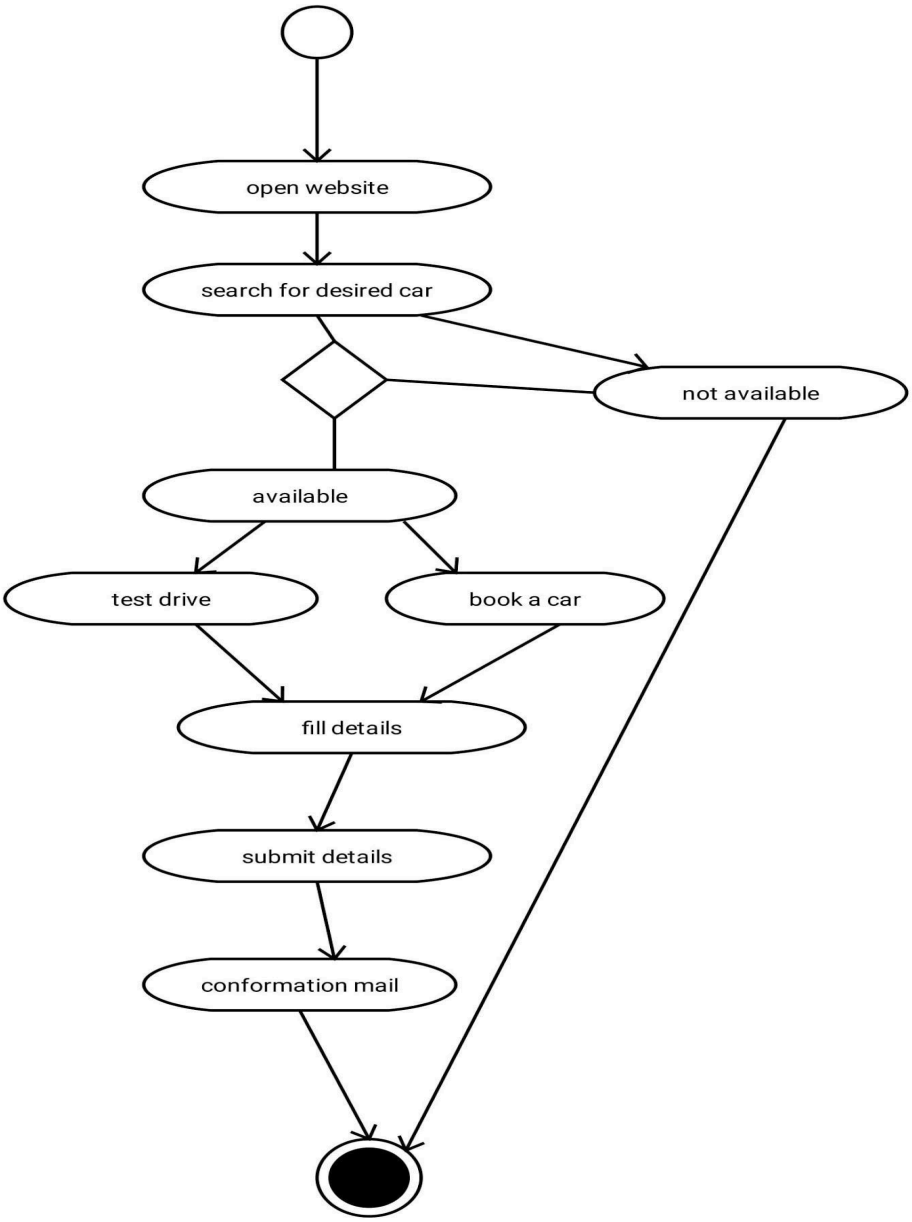


Fig 3: Activity Diagram for online car booking System

3.4 Component Diagram:

Component diagrams are used in modelling the physical aspects of object-oriented systems that are used for visualizing, specifying, and documenting component-based systems and also for constructing executable systems through forward and reverse engineering. Component diagrams are essentially class diagrams that focus on a system's components that often used to model the static implementation view of a system.

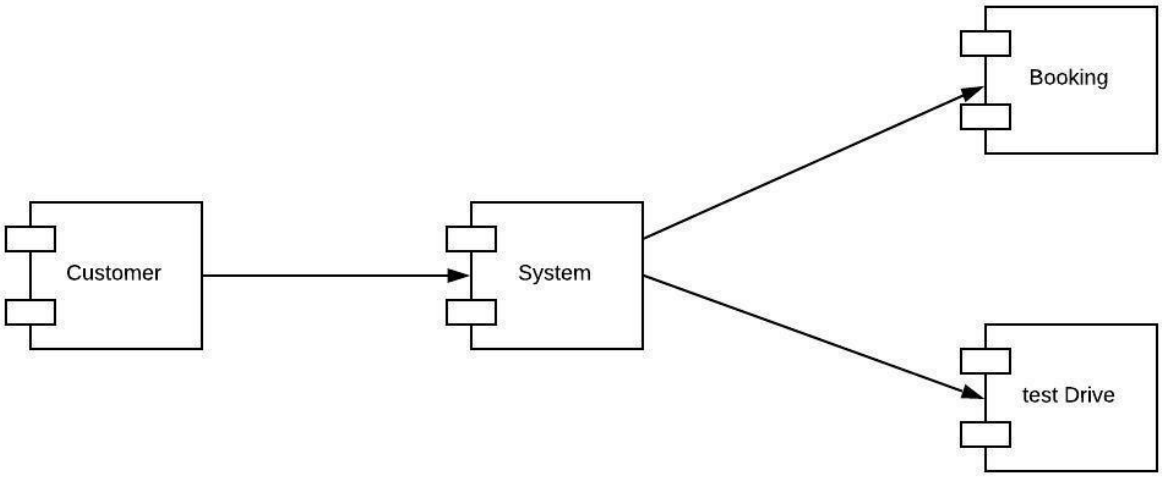


Fig 4: Component Diagram for online car booking System

4. IMPLEMENTATION

4.1HARDWARE DESCRIPTION

4.1.1 Laptop/Desktop

A laptop/desktop is required to use the webapp and to demonstrate all required modules. A **laptop** (also **laptop computer**), often called a **notebook**, is a small, portable personal computer (PC) with a "clamshell" form factor, typically having a thin LCD or LED computer screen mounted on the inside of the upper lid of the clamshell and an alphanumeric keyboard on the inside of the lower lid. The clamshell is opened up to use the computer. Laptops are folded shut for transportation, and thus are suitable for mobile use. Its name comes from lap, as it was deemed to be placed on a person's lap when being used. Although originally there was a distinction between laptops and notebooks (the former being bigger and heavier than the latter), as of 2014, there is often no longer any difference. Today, laptops are commonly used in a variety of settings, such as at work, in education, for playing games, web browsing, for personal multimedia, and general home computer use.

Laptops combine all the input/output components and capabilities of a desktop computer, including the display screen, speakers, a keyboard, data storage device, sometimes an optical disc drive, pointing devices (such as a touchpad or trackpad), with a operating system, a processor and memory into a single unit. Most modern laptops feature integrated webcams and built-in microphones, while many also have touchscreens. Laptops can be powered either from an internal battery or by an external power supply from an AC adapter. Hardware specifications, such as the processor speed and memory capacity, significantly vary between different types, makes, models and price points.

Design elements, form factor and construction can also vary significantly between models depending on intended use. Examples of specialized models of laptops include rugged notebooks for use in construction or military applications, as well as low production cost laptops such as those from the One Laptop per Child (OLPC) organization, which incorporate features like solar charging and semi-flexible components not found on most laptop computers. Portable computers, which later developed into modern laptops, were originally considered to be a small niche market, mostly for specialized field applications, such as in the military, for accountants, or for traveling sales representatives. As the portable computers evolved into the modern laptop, they became widely used for a variety of purposes.

4.1.2 Internet Connectivity

Internet access is the ability of individuals and organizations to connect to the Internet using computer terminals, computers, and other devices; and to access services such as email and the World Wide Web. Internet access is sold by Internet service providers (ISPs) delivering connectivity at a wide range of data transfer rates via various networking technologies. Many organizations, including a growing number of municipal entities, also provide cost-free wireless access and landlines.

Availability of Internet access was once limited, but has grown rapidly. In 1995, only 0.04 percent of the world's population had access, with well over half of those living in the United States,[1] and consumer use was through dial-up. By the first decade of the 21st century, many consumers in developed nations used faster broadband technology, and by 2014, 41 percent of the world's population had access, broadband was almost ubiquitous worldwide, and global average connection speeds exceeded one megabit per second.

The internet connectivity in our webapp is required to send details of the customer to the server and in turn to store them in a database which is accessed by authorized personnel.

4.2. SOFTWARE DESCRIPTION

4.2.1 HTML (Hypertext Markup Language):

HTML is the most basic building block of the Web. It defines the meaning and structure of web content. Other technologies besides HTML are generally used to describe a web page's appearance/presentation (CSS) or functionality/behavior (JavaScript).

"Hypertext" refers to links that connect web pages to one another, either within a single website or between websites. Links are a fundamental aspect of the Web. By uploading content to the Internet and linking it to pages created by other people, you become an active participant in the World Wide Web.

HTML uses "markup" to annotate text, images, and other content for display in a Web browser. HTML markup includes special "elements" such as <head>, <title>, <body>, <header>, <footer>, <article>, <section>, <p>, , , <img>, <aside>, <audio>, <canvas>,

<datalist>, <details>, <embed>, <nav>, <output>, <progress>, <video>, <ul>, <ol>, <li> and many others.

An HTML element is set off from other text in a document by "tags", which consist of the element name surrounded by "<" and ">". The name of an element inside a tag is case insensitive. That is, it can be written in uppercase, lowercase, or a mixture. For example, the <title> tag can be written as <Title>, <TITLE>, or in any other way

HTML elements are the building blocks of HTML pages. With HTML constructs, images and other objects such as interactive forms may be embedded into the rendered page. HTML provides a means to create structured documents by denoting structural semantics for text such as headings, paragraphs, lists, links, quotes and other items. HTML elements are delineated by *tags*, written using angle brackets. Tags such as <**img** /> and <**input** /> directly introduce content into the page. Other tags such as <**p**> surround and provide information about document text and may include other tags as sub-elements. Browsers do not display the HTML tags, but use them to interpret the content of the page.

HTML can embed programs written in a scripting language such as JavaScript, which affects the behavior and content of web pages. Inclusion of CSS defines the look and layout of content. The World Wide Web Consortium (W3C), former maintainer of the HTML and current maintainer of the CSS standards, has encouraged the use of CSS over explicit presentational HTML since 1997.

HTML documents imply a structure of nested HTML elements. These are indicated in the document by HTML *tags*, enclosed in angle brackets thus: <**p**>.

In the simple, general case, the extent of an element is indicated by a pair of tags: a "start tag" <**p**> and "end tag" </**p**>. The text content of the element, if any, is placed between these tags.

Tags may also enclose further tag markup between the start and end, including a mixture of tags and text. This indicates further (nested) elements, as children of the parent element.

The start tag may also include *attributes* within the tag. These indicate other information, such as identifiers for sections within the document, identifiers used to bind style information to the presentation of the document, and for some tags such as the <**img**> used to embed images, the reference to the image resource in the format like this: <**img** src="example.com/example.jpg"

Some elements, such as the line break <**br**>, or <**br** /> do not permit *any* embedded content, either text or further tags. These require only a single empty tag (akin to a start tag) and do not use an end tag.

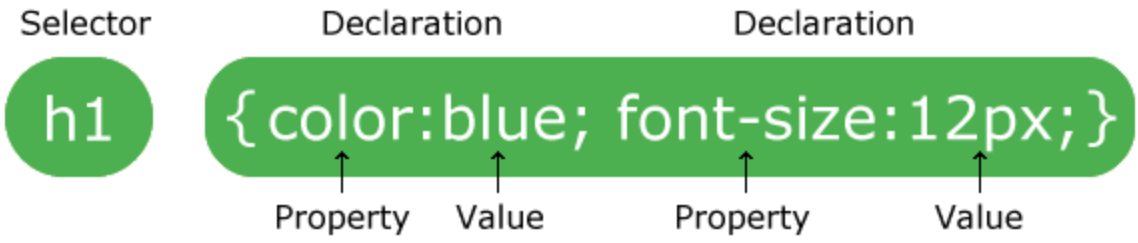
Many tags, particularly the closing end tag for the very commonly used paragraph element <**p**>, are optional. An HTML browser or other agent can infer the closure for the end of an element from the context and the structural rules defined by the HTML standard. These rules are complex and not widely understood by most HTML coders.

The general form of an HTML element is therefore: <**tag** attribute1="value1" attribute2="value2">"content"</**tag**>. Some HTML elements are defined as *empty elements* and take the form <**tag** attribute1="value1" attribute2="value2">. Empty elements may enclose no content, for instance, the <**br**> tag or the inline <**img**> tag. The name of an HTML element is the name used in the tags. Note that the end tag's name is preceded by a slash character, /, and that in empty elements the end tag is neither required nor allowed. If attributes are not mentioned, default values are used in each case.

4.2.2 CSS (Cascading Style Sheet):

Cascading Style Sheets (**CSS**) is a style sheet language used for describing the presentation of a document written in a markup language like HTML.CSS is a cornerstone technology of the World Wide Web, alongside HTML and JavaScript.

CSS is designed to enable the separation of presentation and content, including layout, colors, and fonts. This separation can improve content accessibility, provide more flexibility and control in the specification of presentation characteristics, enable multiple web pages to share formatting by specifying the relevant CSS in a separate .css file, and reduce complexity and repetition in the structural content.



The selector points to the HTML element you want to style.

The declaration block contains one or more declarations separated by semicolons.

Each declaration includes a CSS property name and a value, separated by a colon.

Multiple CSS declarations are separated with semicolons, and declaration blocks are surrounded by curly braces.

CSS selectors are used to "find" (or select) the HTML elements you want to style.

We can divide CSS selectors into five categories:

- Simple selectors (select elements based on name, id, class)
- Combinatory selectors (select elements based on a specific relationship between them)
- Pseudo-class selectors (select elements based on a certain state)
- Pseudo-elements selectors (select and style a part of an element)
- Attribute selectors (select elements based on an attribute or attribute value)

There are three ways of inserting a style sheet:

- External CSS
- Internal CSS
- Inline CSS

External CSS

With an external style sheet, you can change the look of an entire website by changing just one file!

Each HTML page must include a reference to the external style sheet file inside the <link> element, inside the head section.

An external style sheet can be written in any text editor, and must be saved with a .css extension.

The external .css file should not contain any HTML tags.

Internal CSS

An internal style sheet may be used if one single HTML page has a unique style.

The internal style is defined inside the <style> element, inside the head section.

Inline CSS

An inline style may be used to apply a unique style for a single element.

To use inline styles, add the style attribute to the relevant element. The style attribute can contain any CSS property.

4.2.3Python:

Python is a high-level, interpreted, interactive and object-oriented scripting language. Python is designed to be highly readable. It uses English keywords frequently where as other languages use punctuation, and it has fewer syntactical constructions than other languages.

Characteristics of Python:

Following are important characteristics of Python Programming –

- It supports functional and structured programming methods as well as OOP.
- It can be used as a scripting language or can be compiled to byte-code for building large applications.
- It provides very high-level dynamic data types and supports dynamic type checking.
- It supports automatic garbage collection.
- It can be easily integrated with C, C++, COM, ActiveX, CORBA, and Java.
- Python can be used on a server to create web applications.
- Python can be used alongside software to create workflows.
- Python can connect to database systems. It can also read and modify files.
- Python can be used to handle big data and perform complex mathematics.
- Python can be used for rapid prototyping, or for production-ready software development.
- Python works on different platforms (Windows, Mac, Linux, Raspberry Pi, etc).
- Python has a simple syntax similar to the English language.
- Python has syntax that allows developers to write programs with fewer lines than some other programming languages.
- Python runs on an interpreter system, meaning that code can be executed as soon as it is written. This means that prototyping can be very quick.
- Python can be treated in a procedural way, an object-orientated way or a functional way.

Python Syntax compared to other programming languages

- Python was designed for readability, and has some similarities to the English language with influence from mathematics.
- Python uses new lines to complete a command, as opposed to other programming languages which often use semicolons or parentheses.

- Python relies on indentation, using whitespace, to define scope; such as the scope of loops, functions and classes. Other programming languages often use curly-brackets for this purpose.

4.3.4 Flask:

Flask is a micro web framework written in Python. It is classified as a micro framework because it does not require particular tools or libraries.[3] It has no database abstraction layer, form validation, or any other components where pre-existing third-party libraries provide common functions. However, Flask supports extensions that can add application features as if they were implemented in Flask itself. Extensions exist for object-relational mappers, form validation, and upload handling, various open authentication technologies and several common framework related tools. Extensions are updated far more frequently than the core Flask program.

The micro framework Flask is based on the *Pocoo* projects *Werkzeug* and *Jinja2*.

Werkzeug

Werkzeug is a utility library for the Python programming language, in other words a toolkit for Web Server Gateway Interface (WSGI) applications, and is licensed under a BSD License. Werkzeug can realize software objects for request, response, and utility functions. It can be used to build a custom software framework on top of it and supports Python 2.7 and 3.5 and later.[14][15]

Jinja

Main article: Jinja (template engine)

Jinja, also by Ronacher, is a template engine for the Python programming language and is licensed under a BSD License. Similar to the Django web framework, it handles templates in a sandbox.

Features

- Development server and debugger
- Integrated support for unit testing
- RESTful request dispatching
- Uses Jinja templating
- Support for secure cookies (client side sessions)
- 100% WSGI 1.0 compliant
- Unicode-based
- Extensive documentation
- Google App Engine compatibility

- Extensions available to enhance features desired

Flask does not include a database abstraction layer, form validation or anything else where different libraries already exist that can handle that. Instead, Flask supports extensions to add such functionality to your application as if it was implemented in Flask itself. Numerous extensions provide database integration, form validation, and upload handling, various open authentication technologies, and more. Flask may be “micro”, but it’s ready for production use on a variety of needs. The “micro” in micro framework means Flask aims to keep the core simple but extensible. Flask won’t make many decisions for you, such as what database to use.

Flask has many configuration values, with sensible defaults, and a few conventions when getting started. By convention, templates and static files are stored in subdi3 rectories within the application’s Python source tree, with the names templates and static respectively. While this can be changed, you usually don’t have to, especially when getting started.

One of the design decisions in Flask was that simple tasks should be simple; they should not take a lot of code and yet they should not limit you. Because of that, Flask has a few design choices that some people might find surprising or unorthodox. For example, Flask uses thread-local objects internally so that you don’t have to pass objects around from function to function within a request in order to stay thread safe. This approach is convenient, but requires a valid request context for dependency injection or when attempting to reuse code which uses a value pegged to the request. The Flask project is honest about thread-locals, does not hide them, and calls out in the code and documentation where they are used.

The flask script is nice to start a local development server, but you would have to restart it manually after each change to your code. That is not very nice and Flask can do better. If you enable debug support the server will reload itself on code changes, and it will also provide you with a helpful debugger if things go wrong.

4.2.5 Airtable:

Airtable is a cloud collaboration service headquartered in San Francisco. It was founded in 2012 by Howie Liu, Andrew Ofstad, and Emmett Nicholas.

Airtable is a spreadsheet-database hybrid, with the features of a database but applied to a spreadsheet. The fields in an Airtable table are similar to cells in a spreadsheet, but have types such as 'checkbox', 'phone number', and 'drop-down list', and can reference file attachments like images.[1][2]

Users can create a database, set up column types, add records, link tables to one another, collaborate, sort records and publish views to external websites.

Airtable has six basic components:

1. **Bases**: All the information needed to create a project is contained in a Base. Bases can be built from existing templates provided by Airtable. They can also be built from scratch, from a spreadsheet or from an existing Base.
2. **Tables**: A table is similar to a spreadsheet. A base is a collection of tables.
3. **Views**: Views show the result sets of data queries and can be saved for future purposes.
4. **Fields**: Each entry in a table is a field. They are not just restricted to hold text. Airtable currently offers 16 basic field types.[] These are: single-line texts, long text articles, file attachments, check-boxes, single select from drop-down list, multiple-selects from drop-down lists, date and time, phone numbers, email ids, URLs, numbers, currency, percentage, auto-number, formulae and barcodes.
5. **Records**: Each row of a Table is a Record.
6. **Workspaces:** A workspace is a collection of Bases in Airtable.

Linking between tables:

To avoid the need to form a single large table when there is related data in multiple tables, Airtables provides an option to link records of different tables. Airtable allows linking existing tables of related records, creating a new linked table and also multiple links between existing tables.

Collaboration

Airtable allows multiple users to work simultaneously on the same Base allowing more productivity at the workplace. A new collaborator is added by clicking share button found at top of Base and providing the email ID of the collaborator. The owner can set the permission level of collaborator while sharing the Base. There are three permission levels in Airtable Base, namely "Creator", "Edit Only" and "Read Only".

If multiple Bases are required then a Team is formed in Airtable.[] An Airtable Team can hold multiple Bases and collaborators of the Team can work on all the available Bases in the Team. There are four permission levels in Team, unlike for a Base which only has three. The extra permission level in Team is Owner, who has full access to Team Base.

Publishing views

Bases can be easily shared with the public. One need not have an Airtable account to view the published Bases. These view-only Bases shared in public are called Airtable views. These views can be embedded into one's own website and allows the website users see the real-time information in Base, without needing them to have an Airtable account. One possible use-case is Airtable Form, which can be used to take a survey of a product. The feedback received from a customer can also be shown to public by sharing the Airtable View.

Airtable Forms

Airtable introduced Forms in July 2015. Forms could be used to collect data from others like co-workers, customers or the public. A Form can be easily created from an existing Base, and the data collected by the Form is automatically organized in the Airtable Base. Airtable allows user to organize the required fields in the Form. A separate link is created for every Airtable Form. This link can be shared with others to get the required data. An Airtable Form can also be embedded in a website, to get the feedback from the website users.

Airtable integration

Airtable's API can be used to connect to other web services by which information can be exchanged between external web applications and Airtable. Using the Zapier platform, Airtable can connect to over 450 applications and websites. Changes can be set up as triggers for actions in connected applications.

Snapshot

Airtable provides a greater sophistication to back up the data than just allow the users to undo/redo. Airtable periodically snapshots the Base. When a previous version of a Base is needed, the appropriate snapshot is selected by the user from the snapshots list. In addition, the user can manually snapshot a Base at any time.

Airtable provides an API to provide ability to users to build applications that cannot be built within the table constraints. A key, which is present at overview section, is necessary to use

Airtable's API. The key should be kept secret. The API follows REST semantics, uses JSON to encode objects, and uses standard HTTP codes to signal operation outcomes.[20]

Initially, the user has to create a Base in Airtable. Then, Airtable's API can be used to create, read, update, or delete records (i.e. CRUD). As of 2016, Airtable's API does not allow users to create or modify the Base schema.

4.3.6 Bootstrap:

Bootstrap is a framework to help you design websites faster and easier. It includes HTML and CSS based design templates for typography, forms, buttons, tables, navigation, modals, image carousels, etc. It also gives you support for JavaScript plugins.

1. First of all, Bootstrap is the most popular framework for creating layouts. Here are some additional reasons to use Bootstrap:

- Bootstrap's responsive CSS adjusts to phones, tablets, and desktops
- Mobile-first styles are part of the framework
- Bootstrap is compatible with all modern browsers (Chrome, Firefox, Internet Explorer, Safari, and Opera)

2. Bootstrap has a big community and friendly support. For resources visit:

- Check out the Bootstrap Blog.
- You can chat with Bootstrappers with IRC in the irc.freenode.net server, in the ##bootstrap channel.
- Have a look at what people are doing with Bootstrap at the Bootstrap Expo.

3. Bootstrap is easy to set up and create a working layout in less than an hour
They have a basic template available at http://getbootstrap.com/getting-started/#template and also a set of examples for different needs (http://getbootstrap.com/getting-started/#examples). You can just download the bootstrap repository, go to the docs*/examples* folder, copy/paste the example you need and work on it.
4. You don't need to know HTML and CSS well to use bootstrap, it's a plus if you're a backend developer and need to do some UI changes.
5. It's fully customizable, I can choose which components I'd like to use and use variables files to get even more color and behavior customization.

All you need to do is visit http://getbootstrap.com/customize/ , choose the plugins you need and click **download**. Bootstrap also provides a way to override its internal variables for advanced users, but they provide pretty decent defaults, so you shouldn't worry about this unless you need to.
6. When you update the version of Bootstrap, you won't see tons of errors because their core team cares about backwards compatibility.
7. Their documentation is great! Here are some resources to check out:

- Bootstrap 3 Tutorial
- Code Academy: Bootstrap
- Web Design Tutorials: Bootstrap

8. Bootstrap offers a lot of helper classes that make development of a responsive website easy and fast.
You can turn any fixed-width layout into a fluid one by simply changing your parent **.container** class to **.container-fluid**.
Bootstrap also has **.visible-*-*** classes to help you control the way your sections are displayed on tablets and mobile devices. Example:

In this case, the div will be displayed as a section with **display: block** only on phones and tablets. It will be hidden on desktop.
9. Bootstrap's components are well-adopted to the ecosystem of popular JS MVC Frameworks like Angular.

Code Snippets

1. App.py

```python
from flask import Flask, request, render_template
from airtable import Airtable
import smtplib

app = Flask(__name__)
airtable = Airtable('appaVdzZcX4TfXXe8', 'user details', api_key='keyFE90QcwKJZ28UP')
airtable2 = Airtable('appaVdzZcX4TfXXe8', 'bookings', api_key='keyFE90QcwKJZ28UP')
def send_mail(name,data,email_id,car_model,date):
    print('sending email')
    server = smtplib.SMTP('smtp.gmail.com', 587)

    server.ehlo()
    server.starttls()
    server.ehlo()

    server.login("balajitheratipally1212@gmail.com", "vyvrfoazdjphkzcd")

    subject = 'New Registration'
    body = 'Dear {},Thank you for booking a car {} for testdrive  at {} on {} '.format(nam

    message = f'Subject : {subject}\n\n{body}\n'

    server.sendmail('balajitheratipally1212@gmail.com',
                    ['balajitheratipally1212@gmail.com',email_id],
                    message)

    print('email sent')
```

app.py (cont.)

```
@app.route('/')
def home():
    return render_template('video.html')
@app.route('/book',methods=["POST"])
def url_2():
    print(request.form)
    name = request.form['name']
    phone_number = request.form['phone_number']
    email_id = request.form['email_id']

    car_model = request.form['car_model']

    details = {'name': name, 'phone_number': phone_number, 'email_id': email_id,
               'car_model': car_model}

    airtable2.insert(details)

    send_mail2(name,email_id)

    return '<h1>Booking done Successfully..!Thanks for booking.Check your mail for further details..!! </h1>'
@app.route('/<ext>')
def ext_url(ext):
   return render_template(ext)
@app.route('/<ext>',methods=["POST"])
def url_1(ext):
        print(request.form)
        name = request.form['name']
        phone_number = request.form['phone_number']
        email_id = request.form['email_id']
        date = request.form['date']

        car_model = request.form['car_model']

        a=['1AM','2AM','3AM']
        data=''

        x=request.form['options']

        data=data+x+''
        details = {'name': name, 'phone_number': phone_number, 'date': date, 'email_id': email_id,
                'car_model': car_model, 'data': data}

        airtable.insert(details)

        send_mail(name,data,email_id,car_model,date)

        return '<h1>Successfly registered</h1>'
```

2. Registration Page

```
</table><br><br>
<font face="verdana" size=4>
<table class="table table-bordered" style="margin:auto;width:60%">
<tr><td>
<h1 class="text-center"><span class = "label label-default">Want to test Drive..? </span></h1>
<h2>Do you have driving license?</h2>


<div class="container">

<button type="button" class="btn btn-info" data-toggle="collapse" data-target="#demo">Yes</button><br>

<br><br><div id="demo2" ><button type="button" class="btn btn-info" onclick="myFunction()">No</button>

</div>

<script>
function myFunction() {
  var txt;
  if (confirm("sorry you can not register without license")) {
    txt = "test drive option is enabled only for those who have license ";
  } else {
    txt = " Thank you";
  }
  document.getElementById("demo2").innerHTML = txt;
}
</script>

<div id="demo" class="collapse">
<form method="POST">
<label for="name">Enter your name      :</label>
<input type="text" id="name" name="name" value="" required><br>


<label for="phone_number">Enter your phonenumber:</label>
<input type="text" id="phone_number" name="phone_number" value="" required><br>

<label for="email_id">Enter your email id:</label>
<input type="text" id="" name="email_id" value="" required><br>

<label for="date">Enter date           :</label>
<input type="text" id="date" name="date" value="" required><br>

<label for="date">Enter car model      :</label>
<input type="text" id="car_model" name="car_model" value="" required><br>

<mark><h3>Select any one slot</h3></mark>
<h4>Available Slots</h4>
<input type="radio" name="options" id="1AM" value="1AM">1AM</input><br>
<input type="radio" name="options" id="2AM" value="2AM">2AM</input><br>
<input type="radio" name="options" id="3AM" value="3AM">3AM</input><br>
<button onclick= " ">Submit</button>
</form>
```

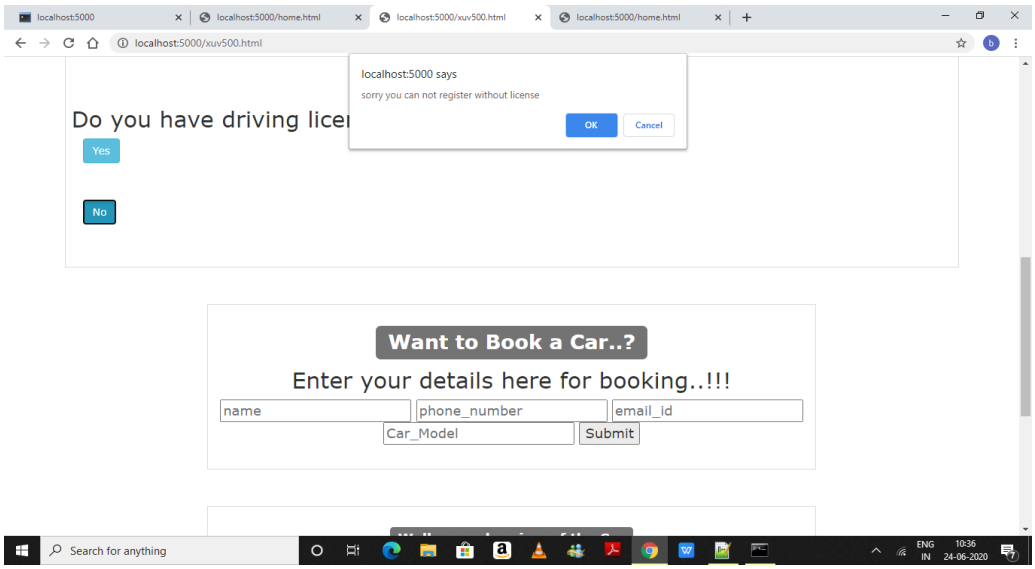
3. Home Page

```
<html>
<head>

</head>


<body bgcolor="yellow">
<div align="center">
<fieldset id="fs"><legend id="lg" align=center>TYPES OF CARS AVAILABLE</legend>
    <marquee behavior="alternate">********HURRAY********</marquee>
<marquee behavior="alternate">BUY A CAR AT LOW PRICE</marquee>
<font face="verdana" size=4>
<table>
<tr><td>
<ol>

<li><a href="hatchback.html" target="centre">HATCH_BACK CARS<br></a></li><br>
<br><li><a href="suv.html" target="centre">SUV</a></li><br>
<br><li><a href="crossover.html" target="centre">CROSSOVER</a></li><br>
<br><li><a href="coupe.html" target="centre">COUPE</a></li><br>
<br><li><a href="minivans.html" target="centre">MINI VANS</a></li><br>
</ol>
</td></tr>
</table>
</font>
</fieldset>
</div>
</body>
</html>
```

TESTING

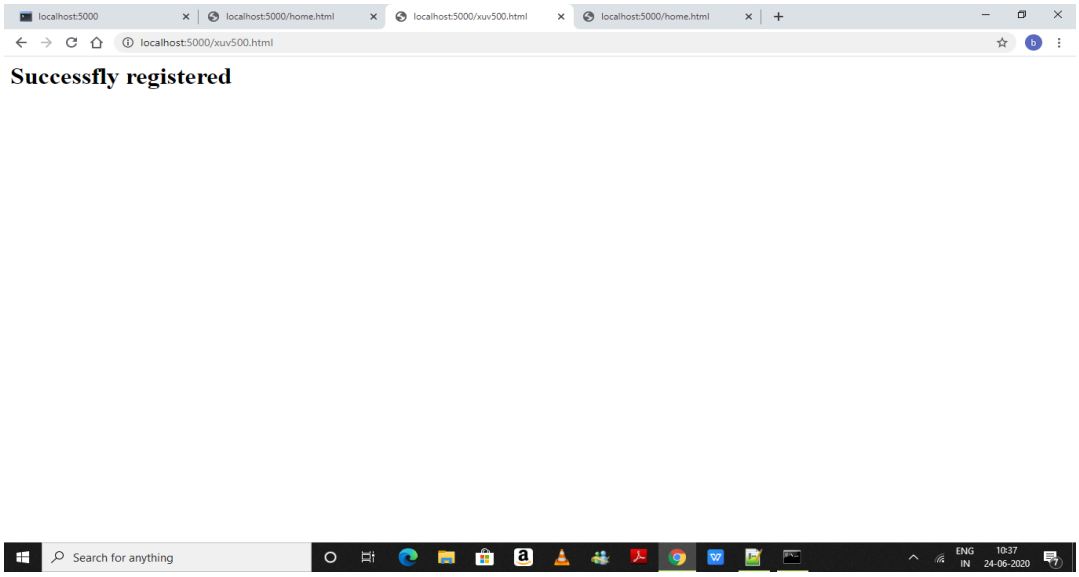
1. No license

This is the output if customer does not have driving license



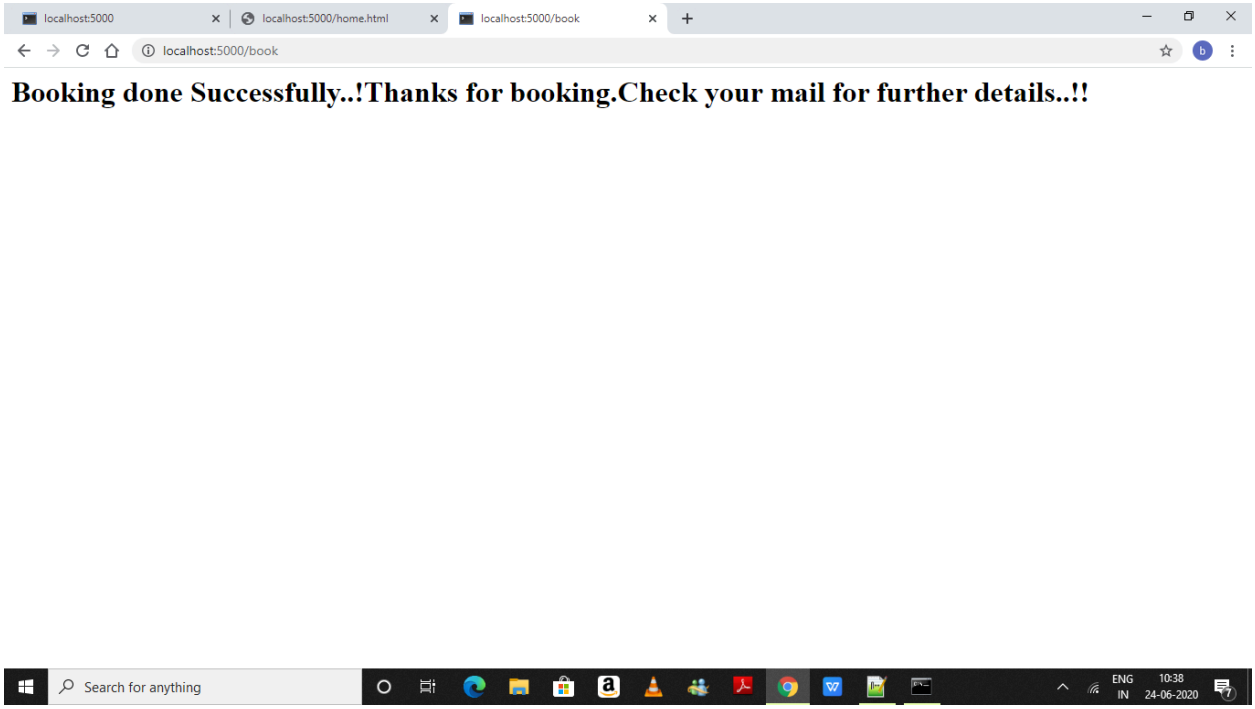
2. Successful Registration

This is the output after registering for test drive online by submitting your details



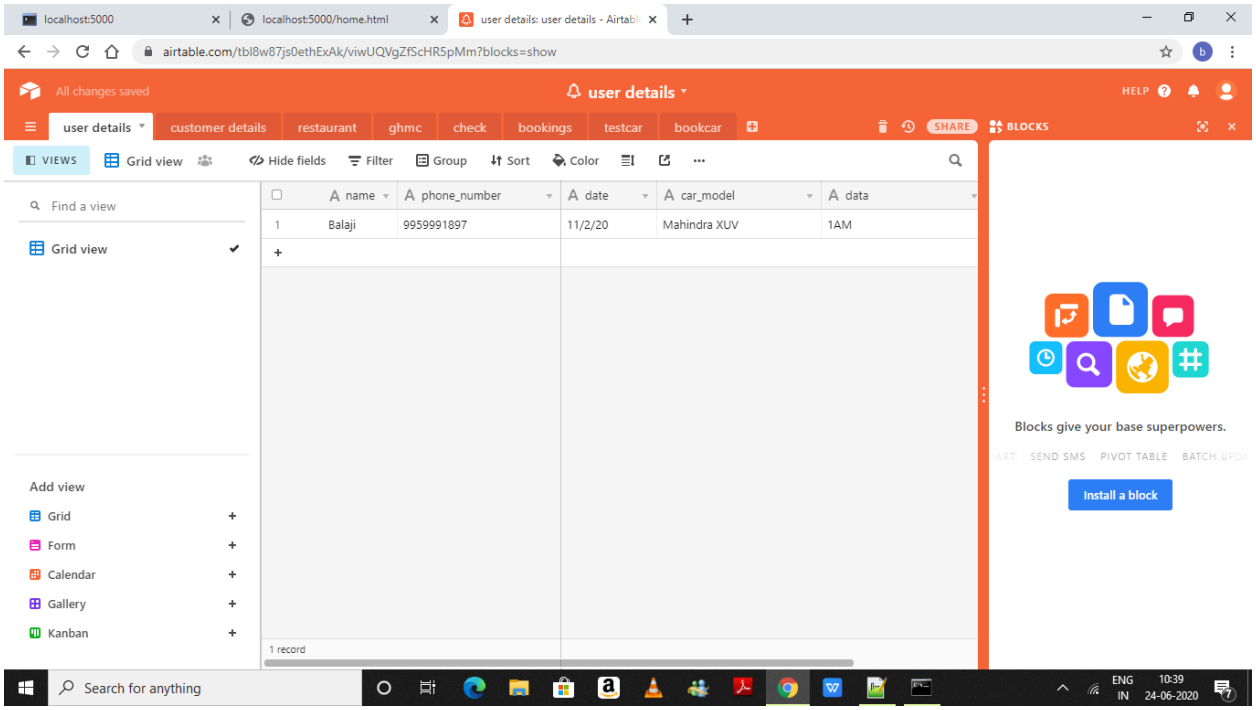
3. Successful Booking

This is the output after booking your dream car.



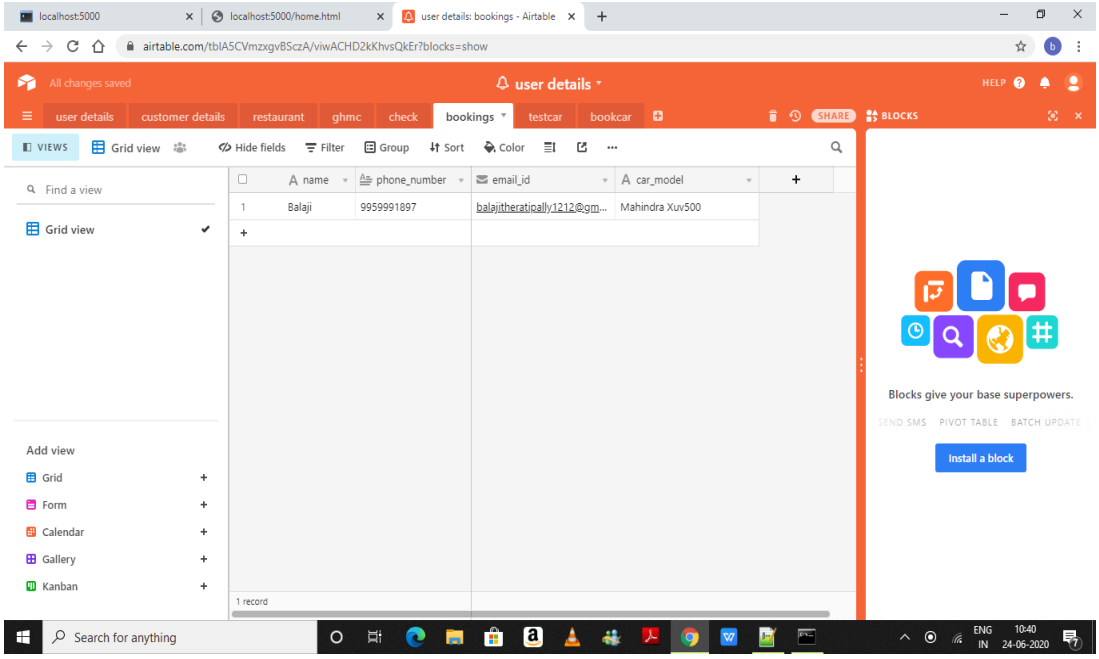
4. Airtable details

This is the output that shows the details of customer stored in airtable.



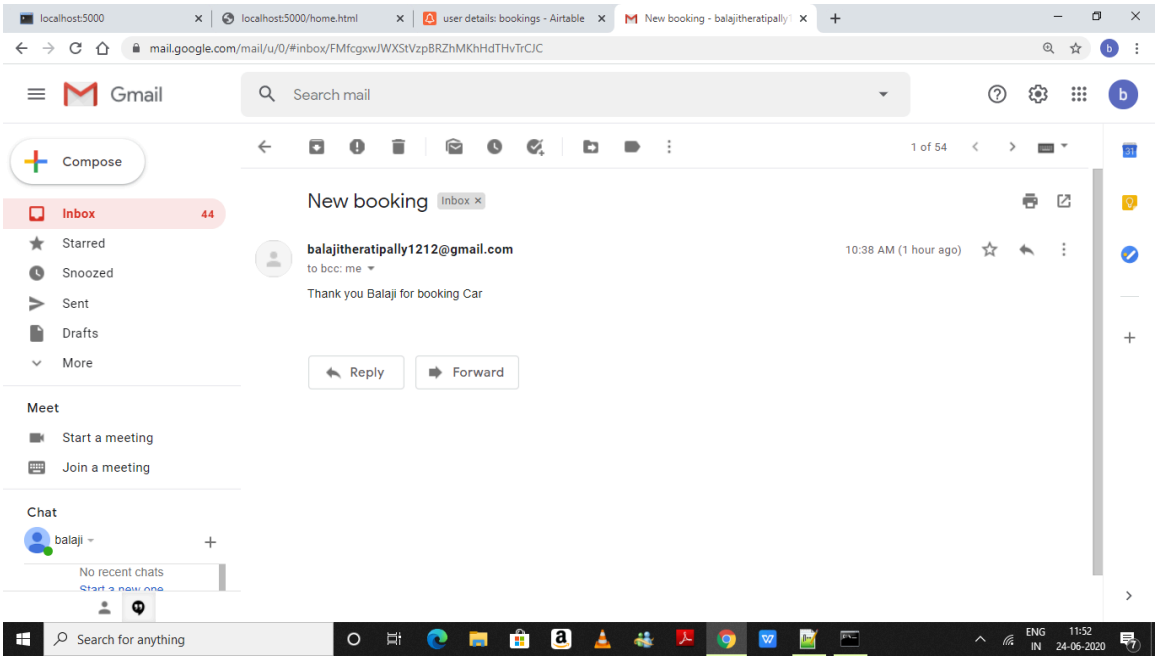
5. Airtable details

This is the output that shows details of customers who have booked a car.



6. Confirmation Mail

This is the output that shows confirmation mail.



Conclusion

The entire application provides a wide scope which satisfies the prevailing demand in the present market scenario in which a customer needs are satisfied. A real-life, demanding application is selected as reference to guide most of the solutions exploration and the implementation decisions. Airtable is a spreadsheet-database hybrid, with the features of a database but applied to a spreadsheet. It can be easily shared with public and is easily maintained and produces more productivity. It allows multiple users to work simultaneously. Flask is the web framework used. Flask can be used for building complex websites. Flask depends on the Jinja template engine and the Werkzeug WSGI toolkit. Flask provides user with tools, libraries and technologies that allow you to build a complex web application. We would like to conclude by saying that this project is used for many purposes. It is also helpful for the future generations.

Future Scope

- This project is mainly used for personal purposes. It helps the customers to know when to go to the showroom without wasting their time.
- The project about online car booking is helpful to know about the various cars available in the showroom and also to either book them or to book a slot for their test drive.

References

1. https://youtu.be/BczLWImAmBk
2. https://youtu.be/9MHYHgh4jYc
3. https://youtu.be/AEM8_4NBU04
4. https://youtu.be/YziPKikI_gM.
5. https://youtu.be/U5MBYN6an70
6. Violent Python: A Cookbook for Hackers, Forensic Analysts, Penetration Testers and Security Engineers
7. NoSQL and SQL Data Modeling: Bringing Together Data, Semantics, and Software First Edition
8. https://airtable.com/
9. https://flask.palletsprojects.com/en/1.1.x/
10. https://www.w3schools.com/bootstrap/

Appendix A

Abbreviations

- HTML:Hyper Text Markup Language
- CSS: Cascading Style Sheets
- IDE: Integrated Development Environment
- USB: Universal Serial Bus
- CDN:Content Delivery Network

Appendix B

Software Installation Process

Step 1: System Requirements:

You will be delighted, to know that you can start your project on either of the following operating systems:

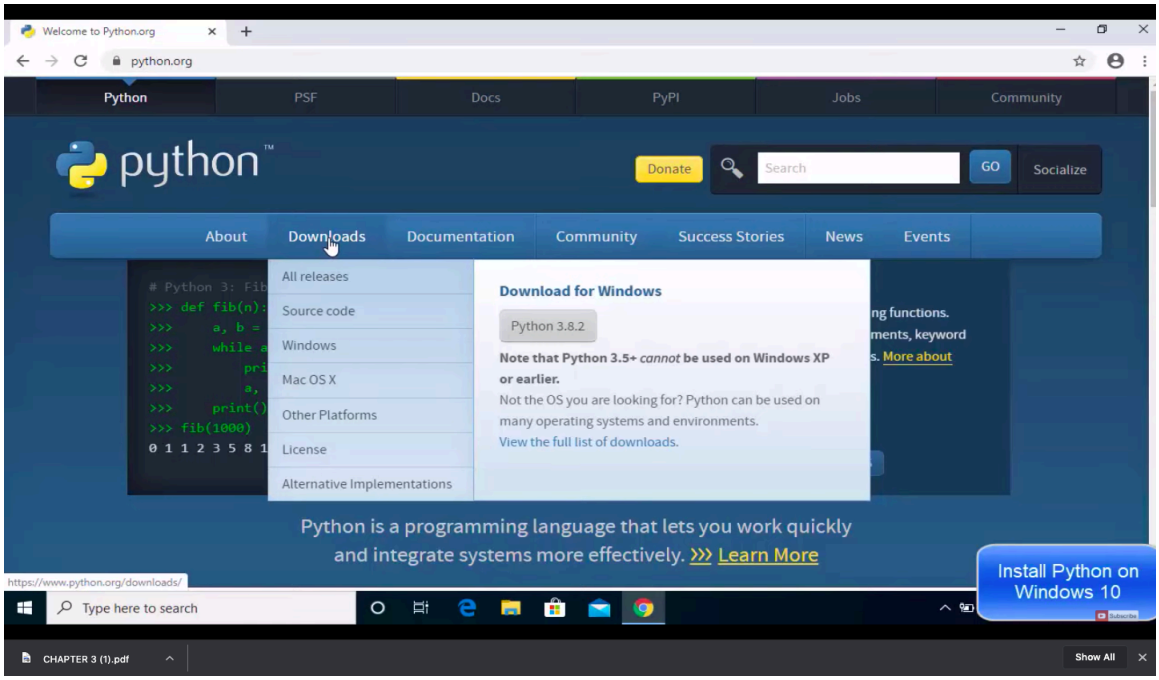
- Microsoft Windows 10/8/7/Vista.
- Mac OS X 10.8.5 or higher.
- Ubuntu.

Second point is that all the required tools to develop your project are open source and can be downloaded from the Web. Following is the list of software's you will need before you start your project.

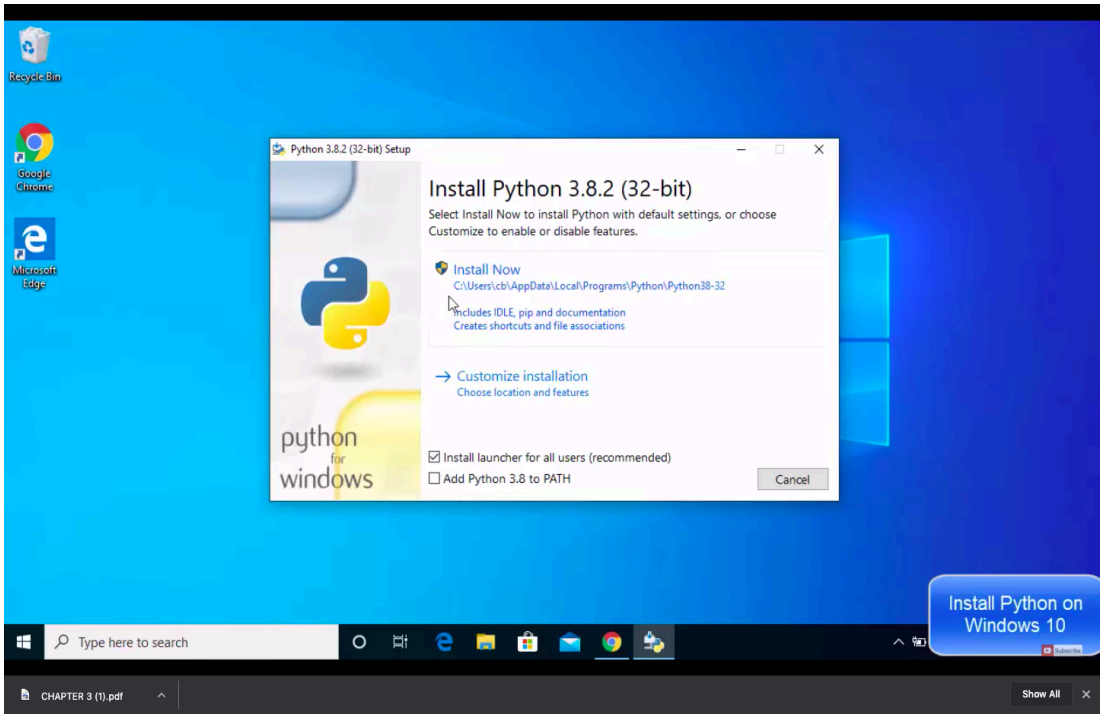
- Python.
- Airtable.
- Flask

Step 2: Setup Python

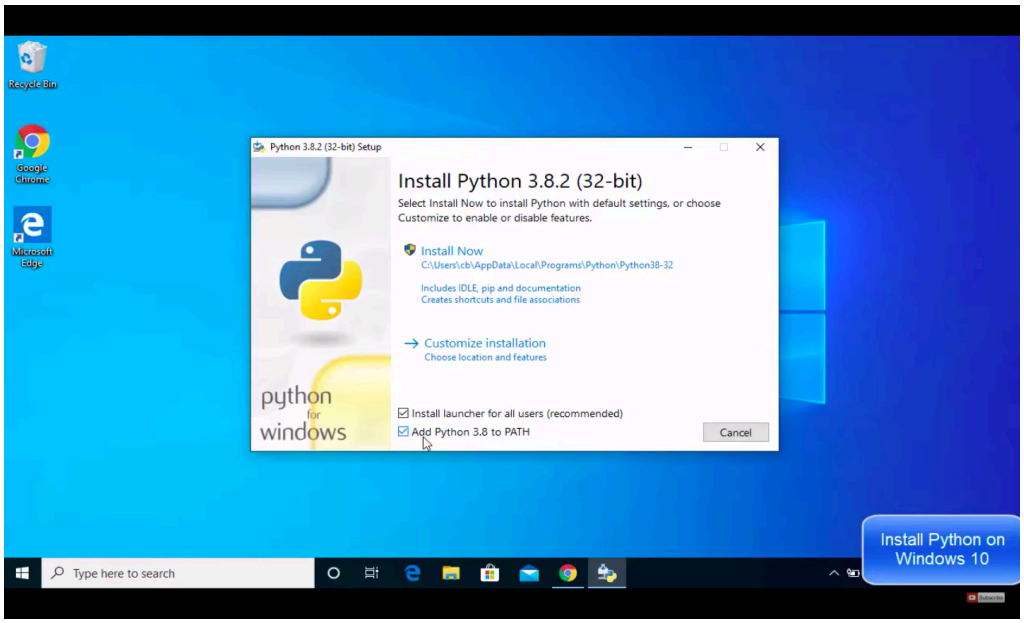
- Download the Python.



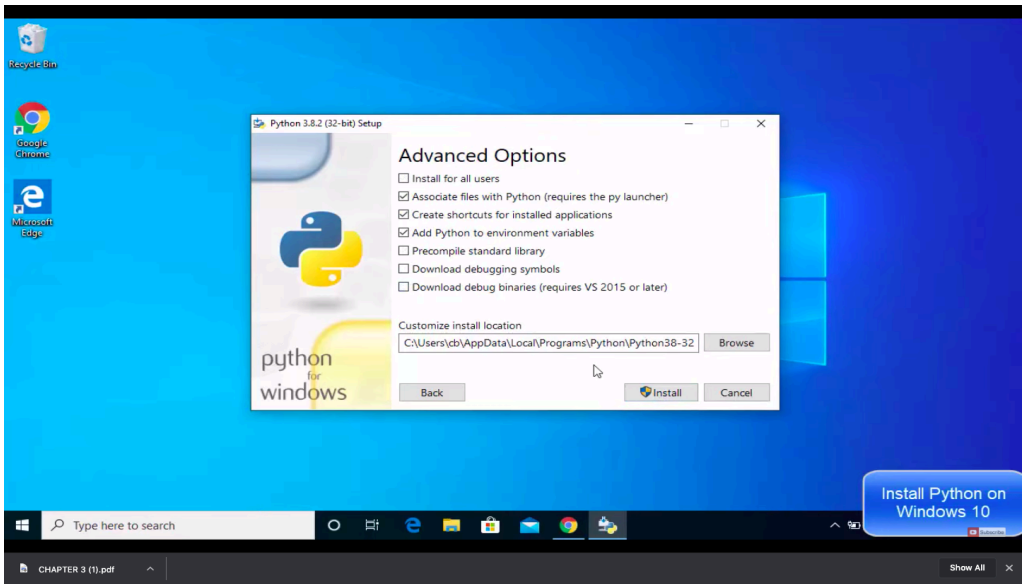
When the file has downloaded, install the contents onto the Desktop.

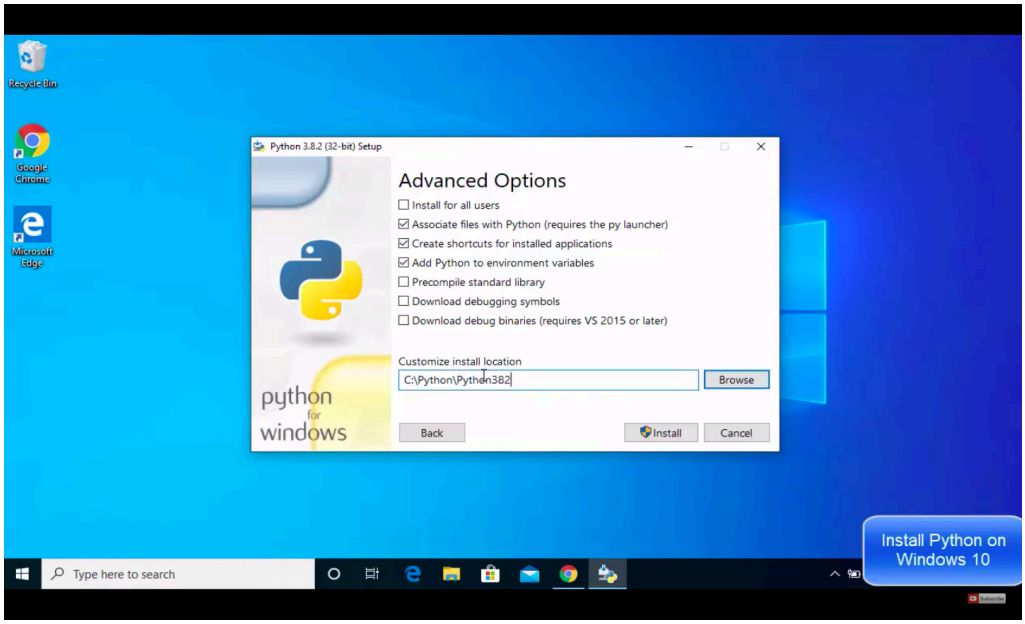


Select add python to PATH



Next select your installation location





Now your python setup is complete

Check whether python is successfully installed or not

```
C:\Users\Dejan>python --version
Python 3.7.3
```

The next step is to install Flask.

Step 1: To create an environment

Create a project folder and a `venv` folder within:

```
$ mkdir myproject
$ cd myproject
$ python3 -m venv venv
```

On Windows:

```
$ py -3 -m venv venv
```

If you needed to install virtualenv because you are using Python 2, use the follov

```
$ python2 -m virtualenv venv
```

On Windows:

```
> \Python27\Scripts\virtualenv.exe venv
```

Step 2: To activate the environment

Before you work on your project, activate the corresponding environment:

```
$ . venv/bin/activate
```

On Windows:

```
> venv\Scripts\activate
```

Your shell prompt will change to show the name of the activated environment.

Step 3: To install flask

```
$ pip install Flask
```

Your environment is set and is ready to use.

Appendix C

Software Usage Process

Python is a general purpose and high level programming language. You can use Python for developing desktop GUI applications, websites and web applications. Also, Python, as a high level programming language, allows you to focus on core functionality of the application by taking care of common programming tasks. The simple syntax rules of the programming language further makes it easier for you to keep the code base readable and application maintainable.

7 Reasons Why One Must Consider Writing Software Applications in Python

1) Readable and Maintainable Code

While writing a software application, you must focus on the quality of its source code to simplify maintenance and updates. The syntax rules of Python allow you to express concepts without writing additional code. At the same time, Python, unlike other programming languages, emphasizes on code readability, and allows you to use English keywords instead of punctuations. Hence, you can use Python to build custom applications without writing additional code. The readable and clean code base will help you to maintain and update the software without putting extra time and effort.

2) Multiple Programming Paradigms

Like other modern programming languages, Python also supports several programming paradigm. It supports object oriented and structured programming fully. Also, its language features support various concepts in functional and aspect-oriented programming. At the same time, Python also features a dynamic type system and automatic memory management. The programming paradigms and language features help you to use Python for developing large and complex software applications.

3) Compatible with Major Platforms and Systems

At present, Python is supports many operating systems. You can even use Python interpreters to run the code on specific platforms and tools. Also, Python is an interpreted programming language. It allows you to you to run the same code on multiple platforms without recompilation. Hence, you are not required to recompile the code after making any alteration. You can run the modified application code without recompiling and check the impact of changes made to the code immediately. The feature makes it easier for you to make changes to the code without increasing development time.

4) Robust Standard Library

Its large and robust standard library makes Python score over other programming languages. The standard library allows you to choose from a wide range of modules according to your precise needs. Each module further enables you to add functionality to the Python application without writing additional code. For instance, while writing a web application in Python, you can use specific modules to implement web services, perform string operations, manage operating system interface or work with internet protocols. You can even gather information about various modules by browsing through the Python Standard Library documentation.

5) Many Open Source Frameworks and Tools

As an open source programming language, Python helps you to curtail software development cost significantly. You can even use several open source Python frameworks, libraries and development tools to curtail development time without increasing development cost. You even have option to choose from a wide range of open source Python frameworks and development tools according to your precise needs. For instance, you can simplify and speedup web application development by using robust Python web frameworks like Django, Flask, Pyramid, Bottle and Cherrypy. Likewise, you can accelerate desktop GUI application development using **Python GUI frameworks** and toolkits like PyQT, PyJs, PyGUI, Kivy, PyGTK and WxPython.

6) Simplify Complex Software Development

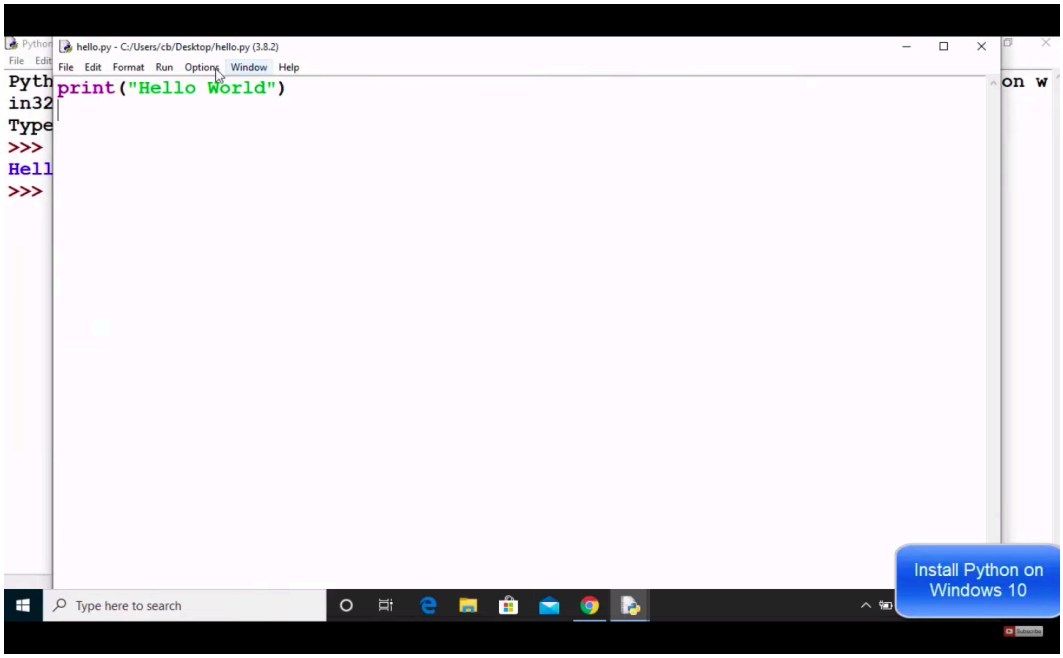
Python is a general purpose programming language. Hence, you can use the programming language for developing both desktop and web applications. Also, you can use Python for developing complex scientific and numeric applications. Python is designed with features to facilitate data analysis and

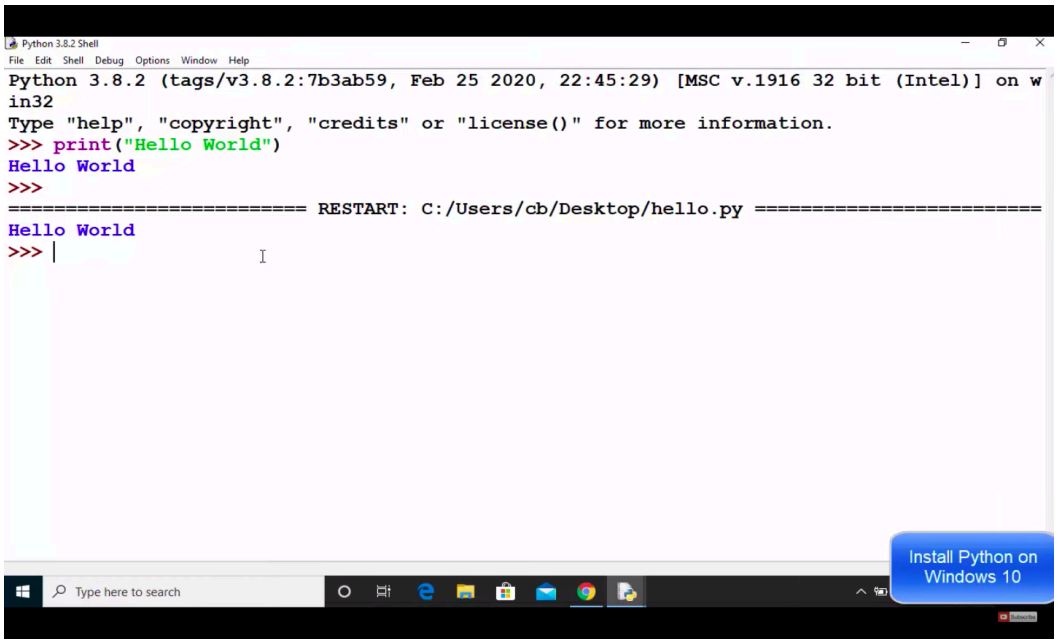
visualization. You can take advantage of the data analysis features of Python to create custom big data solutions without putting extra time and effort. At the same time, the data visualization libraries and APIs provided by Python help you to visualize and present data in a more appealing and effective way. Many **Python developers** even use Python to accomplish artificial intelligence (AI) and natural language processing tasks.

7) Adopt Test Driven Development

You can use Python to create prototype of the software application rapidly. Also, you can build the software application directly from the prototype simply by refactoring the Python code. Python even makes it easier for you to perform coding and testing simultaneously by adopting test driven development (TDD) approach. You can easily write the required tests before writing code and use the tests to assess the application code continuously. The tests can also be used for checking if the application meets predefined requirements based on its source code.

However, Python, like other programming languages, has its own shortcomings. It lacks some of the built-in features provided by other modern programming language. Hence, you have to use Python libraries, modules, and frameworks to accelerate custom software development. Also, several studies have shown that Python is slower than several widely used programming languages including Java and C++. You have to speed up the Python application by making changes to the application code or using custom runtime. But you can always use Python to speed up software development and simplify software maintenance.





Demo of Python usage in IDE