Table of Contents

Introduction

Hotel Management System is a system that provides us to reserving rooms, checking whether the rooms are vacant are or not etc. by using bon line browsing. This system is very useful to all especially for business people. For Business people they don't have sufficient time for these then they can use these type of online Hotel Management Systems. By this project we will reduce the faults in bills of their expenditure and decrease time of delay to give the bills to the customers. We can also save the bills of the customer. By this project we can also include all the taxes on the bills according to their expenditures. It has a scope to reduce the errors in making the bills. Computerized bill can be printed within fraction of seconds. Online ordering of Booking is possible by using this software. This Project is based on php. If any one wants to book the room for few days then they can specify the specific number by seeing the types of rooms we have. The bill of this online booking is based on the type of room they can select is displayed. HOTEL MANAGEMENT SYSTEM is a hotel reservation site script where site users will be able to search rooms availability with an online booking reservations system. Site users can also browse hotels, view room inventory, check availability, and book reservations in real-time. Site users enter check in date and check out date then search for availability and rates. After choosing the right room in the wanted hotel–all booking and reservation process is done on the site and an SMS is sent to confirm the booking. The main objective of the hotel administration is to maintain a constant inflow of visitors and guests throughout the year, as well as promoting the wide range of hotel services and USPS and how to benefit from customers who visit through campaigns of marketing. This application allows the hotel's administration the possibility of using the entire system from a single online interface, giving them more power and flexibility. Booking The rooms, staff management, and other hotel management services are included in this project. The manager can use the system to advertise the available rooms. Customers can see and book rooms from the comfort of their homes. The administrator has the authority to approve or refuse a request for a client booking. Other hotel services are also available for consumers to see and book. As a result, the system can be used both by consumers and the administration to control the activities of the hotel in motion.

Features

- Fully responsive website(for both mobile and larger screens) based on google material design.

- Login/Signup/Logout feature for both Customer and Room Manager.
- Custom dashboard for both Customer and Room Manager.
- Facility to add, delete, update rooms by Room Manager.
- The room Manager can see the details of the user that has booked one of his rooms.
- Customers can cancel the room booking.
- Contact form support for every visitor of the website.
- Superusers have access to all the functionality listed above.

Implementation

Prepare Your Environment

When you're ready to start your new Django web application, create a new folder and navigate into it. In this folder, you'll set up a new virtual environment using your command line:

shell

```
$ python3 -m venv env
```

This command sets up a new virtual environment named env in your current working directory. Once the process is complete, you also need to activate the virtual environment:

shell

```
$ source env/bin/activate
```

If the activation was successful, then you'll see the name of your virtual environment, (env), at the beginning of your command prompt. This means that your environment setup is complete.

Install Django and Pin Your Dependencies

Once you've created and activated your Python virtual environment, you can install Django into this dedicated development workspace:

shell

```
(env) $ python -m pip install django
```

This command fetches the Django package from the Python Package Index (PyPI) using pip. After the installation has been completed, you can pin your dependencies to make sure that you're keeping track of which Django version you installed:

shell

```
(env) $ python -m pip freeze > requirements.txt
```

This command writes the names and versions of all external Python packages that are currently in your virtual environment to a file called requirements.txt. This file will include the Django package and all of its dependencies.

Suppose you're working on an existing project with its dependencies already pinned in a requirements.txt file. In that case, you can install the right Django version as well as all the other necessary packages in a single command:

shell

```
(env) $ python -m pip install -r requirements.txt
```

The command reads all names and versions of the pinned packages from your requirements.txt file and installs the specified version of each package in your virtual environment.

Keeping a separate virtual environment for every project allows you to work with different versions of Django for different web application projects. Pinning the dependencies with pip freeze enables you to reproduce the environment that you need for the project to work as expected.

Set Up a Django Project

After you've successfully installed Django, you're ready to create the scaffolding for your new web application. The Django framework distinguishes between **projects** and **apps**:

- **A Django project** is a high-level unit of organization that contains logic that governs your whole web application. Each project can contain multiple apps.
- **A Django app** is a lower-level unit of your web application. You can have zero to many apps in a project, and you'll usually have at least one app. You'll learn more about apps in the next section.

With your virtual environment set up and activated and Django installed, you can now create a project:

shell

```
(env) $ django-admin startproject <project-name>
```

We are going to use hotelsystem as the name of our project.

shell

```
(env) $ django-admin startproject hotelsystem .
```

Note that we add the period (point) at the end of our project. The reason for this is for our project to be created in the base directory without creating a directory or an additional folder.

Running this command creates your project folder structure, which includes some Python files.

```
├── hotelsystem/
│   ├── __init__.py
│   ├── asgi.py
│   ├── settings.py
│   ├── urls.py
│   └── wsgi.py
│
└── manage.pyI
```

n the code block above, you can see the folder structure that the startproject command created for you:

- **hotelsystem/** is your top-level project folder. We have some python files that come as your project subfiles that you'll edit when you work on your web application.
- **manage.py** is a Python file that serves as the command center of your project. It does the same as the Django-admin command-line utility.

Start a Django App

Every project you build with Django can contain multiple Django apps. When you ran the startproject command in the previous section, you created a management app that you'll need for every default project that you'll build. Now, you'll create a Django app that'll contain the specific functionality of your web application.

You don’t need to use the Django-admin command-line utility anymore, and you can execute the startapp command through the manage.py file instead:

shell

```
(env) $ python manage.py startapp <appname>
```

The startapp command generates a default folder structure for a Django app. For this project, we shall create the following apps: api, authentication, booking, payment, customers, and receptionist. So let’s start by building api.

shell

```
(env) $ python manage.py startapp api
```

Once the startapp command has finished execution, you’ll see that Django has added another folder to your folder structure

```
├── api/
│   │
│   ├── migrations/
│   │   └── __init__.py
│   │
│   ├── __init__.py
│   ├── admin.py
│   ├── apps.py
│   ├── models.py
│   ├── tests.py
│   └── views.py
│
├── hotelsystem/
│   ├── __init__.py
│   ├── asgi.py
│   ├── settings.py
│   ├── urls.py
│   └── wsgi.py
│
└── manage.py
```

Run the same command for the creation of the rest of the application Here are three notable files that were created in the app folder:

- __init__.py: Python uses this file to declare a folder as a package, which allows Django to use code from different apps to compose the overall functionality of your web application. You probably won't have to touch this file.
- models.py: You'll declare your app's models in this file, which allows Django to interface with the database of your web application.
- views.py: You'll write most of the code logic of your app in this file.

Go to your project(inside hotelsystem folder) settings.py and register all the apps. see snippet below

```
INSTALLED_APPS = [
'django.contrib.admin',
'django.contrib.auth',
'django.contrib.contenttypes',
'django.contrib.sessions',
'django.contrib.messages',
'django.contrib.staticfiles',
'authentication',
'booking',
'Receptionist',
'customer',
'rest_framework',
'payment',
]
```

Now let's look at the operations of the subfiles of each of the following project apps..

Authentication

An authentication backend is a class that implements two required methods: get_user(user_id) and authenticate(request, **credentials) , as well as a set of optional permission-related authorization methods.

Django provides several views that you can use for handling login, logout, and password management

```
├── authentication/
|   |
|   ├── migrations/
|   |   └── __init__.py
|   |
|   ├── __init__.py
|   ├── admin.py
|   ├── apps.py
|   ├── models.py
|   ├── tests.py
|   └── views.py
```

Advantages and Disadvantages

Advantages:

- Sometimes it happens that the rooms get booked soon when one visits the place therefore user can make advance booking using this system.
- It saves user time in search of rooms.
- The system is useful as it calculates an exact cost for requested number of days
- It saves organization resources and expenses. This system is effective and saves time and cost of users.
- Easy registration.

Disadvantages:

- The booking process usually requires a customer identity which the system cannot detect.
- It requires a reliable internet connection.

Applications:

- This system can be applied in hotels.
- It can also be implemented in resorts.

Future Work

Python is an excellent choice for developing and improving hotel management systems (HMS) due to its versatility, extensive libraries, and support for various technologies. Here's a look at how Python could be used to shape the future of HMS:

- **Personalized Recommendations**: Use Python libraries like TensorFlow or Scikit-learn to build machine learning models that analyze guest preferences and behaviors, enabling personalized recommendations for rooms, services, and activities.
- **Cloud Storage**: Use cloud services for scalable data storage and management, integrating with Python libraries for efficient data access and manipulation.
- **Business Intelligence**: Utilize Python libraries such as Pandas, NumPy, and Matplotlib for data analysis and visualization, helping hotels make data-driven decisions about operations and marketing.
- **Revenue Management**: Build advanced analytics tools to optimize pricing and revenue management strategies by analyzing market trends and guest behavior.
- **APIs for Mobile Apps**: Develop RESTful APIs using Python frameworks like Flask or Django to connect with mobile apps, enabling features like mobile check-in, room key access, and real-time notifications.
- **Web Interfaces**: Use Python web frameworks to create intuitive web interfaces for managing reservations, guest profiles, and hotel operations.
- **Secure Transactions**: Implement blockchain-based solutions for secure payment processing and data management using Python libraries like PyCryptodome.
- **Smart Contracts**: Develop smart contracts to automate and secure booking and payment processes using Python-based blockchain platforms.
- **VR Tours**: Use Python to create or integrate with VR systems that offer virtual tours of hotel rooms and amenities.
- **AR Enhancements**: Develop Python-based AR applications that provide interactive features and information overlays for guests.

Conclusion

In conclusion, using Python to develop and enhance a hotel management system (HMS) offers numerous advantages and opportunities for innovation. Python's versatility, extensive libraries, and strong community support make it an ideal choice for creating a robust, scalable, and efficient HMS. Here's a summary of key points:

Python supports a wide range of functionalities from web development to data analysis and machine learning. This allows for the creation of an all-in-one HMS that can handle reservations, guest management, and operational tasks. Python can seamlessly integrate with various technologies, such as IoT for smart room management, cloud services for scalability, and blockchain for secure transactions. Python's rich ecosystem of libraries and frameworks accelerates development and reduces time-to-market for new features and improvements. The active Python community provides valuable resources, support, and continuous updates, ensuring that HMS solutions remain cutting-edge and reliable.

In summary, Python's capabilities make it a powerful tool for developing modern, efficient, and innovative hotel management systems. By leveraging Python, hotels can enhance their operations, improve guest experiences, and stay competitive in a rapidly evolving industry.

References

1.Python Official Documentation: Python 3.x Documentation. https://docs.python.org/3/
2. pytz Library Documentation: https://pypi.org/project/pytz/
3. Conversion Factors: o National Institute of Standards and Technology (NIST) https://www.nist.gov/ o Wikipedia: Conversion of Units https://en.wikipedia.org/wiki/Conversion_of_units