OPEN SOURCE PROGRAMMING

DIGITAL ASSIGNMENT 1

NAME – PRANEEL MISHRA REGISTRATION NUMBER – 18BIT0426

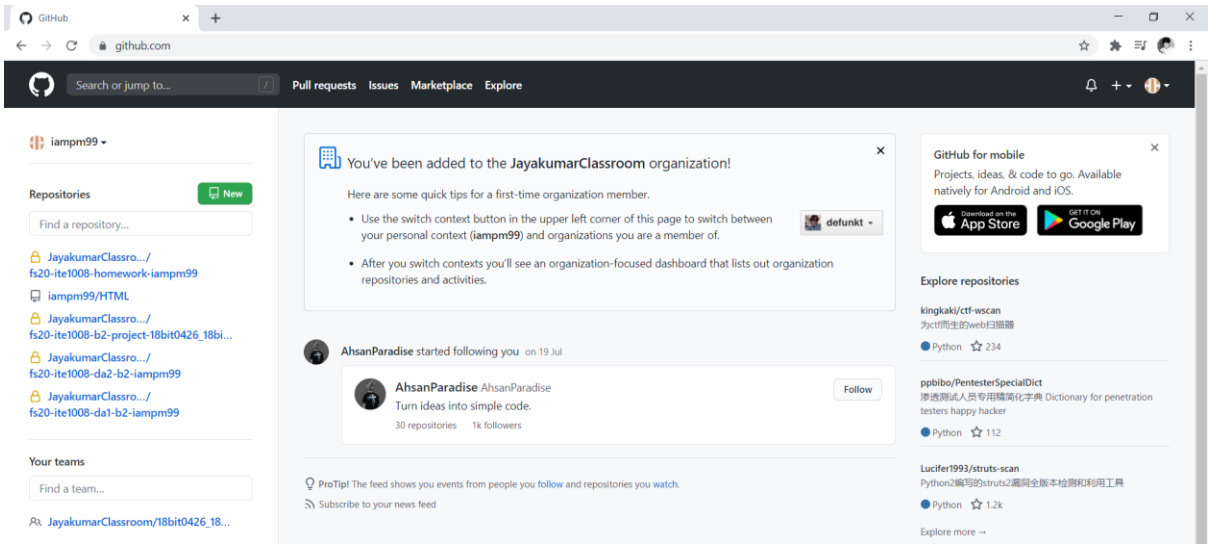
FACULTY – PROF. JAYAKUMAR S. COURSE CODE – ITE1008

STEP BY STEP PROCESS OF GITHUB WORKING METHODOLOGY

- **Creating A Branch :**
  Whenever you are going to work on a project, you will be working on different ideas an the project may have different types of functions. Branching helps to manage these functions all together and keeps a track of each.
  Branching is a core concept in Git, and the entire GitHub flow is based upon it. There's only one rule: anything in the main branch is always deployable.
- **Add Commits :**
  Whenever you add, delete or edit a file in any repository it is known as commit. Commits are basically the history of your work that other people can look upon and understand step by step.
  It also helps to overcome if any error is there in your new commit file ,you can simple revert back to the previous one.
- **Pull Request :**
  You can simple give access to someone to get your files and make changes to it by generating them a pull request when you are in a stuck situation or getting an error By using GitHub's @mention system in your Pull Request message, you can ask for feedback from specific people or teams, whether they're down the hall or ten time zones away.
  Useful for Open Source Project.
- **Discuss And Review Of Code :**
  Once a Pull request has been Opened, the people can see your code and review it and they might have any question regarding it. So they can simply make changes in it and PUSH it back to the repository (Commit) Hence making it easier for the developer. GitHub will show your new commits and any additional feedback you may receive in the unified Pull Request view.
- **Deploy :**
  Deploying is similar or same as Publishing. When the developer is ready with his work and confident enough to put his work worldwide he/she deploys his/her work in Github. Different teams may have different deployment strategies. For some, it may be best to deploy to a specially provisioned testing environment. For others, deploying directly to production may be the better choice based on the other elements in their workflow.
- **Merge :**
  Now if your work is verified by them, you can simply commit to the main branch. Once merged, Pull requests create a history of commits which you might have made while making the code work ,hence, making it easier for other developers to see your work and understand from where it all arose.

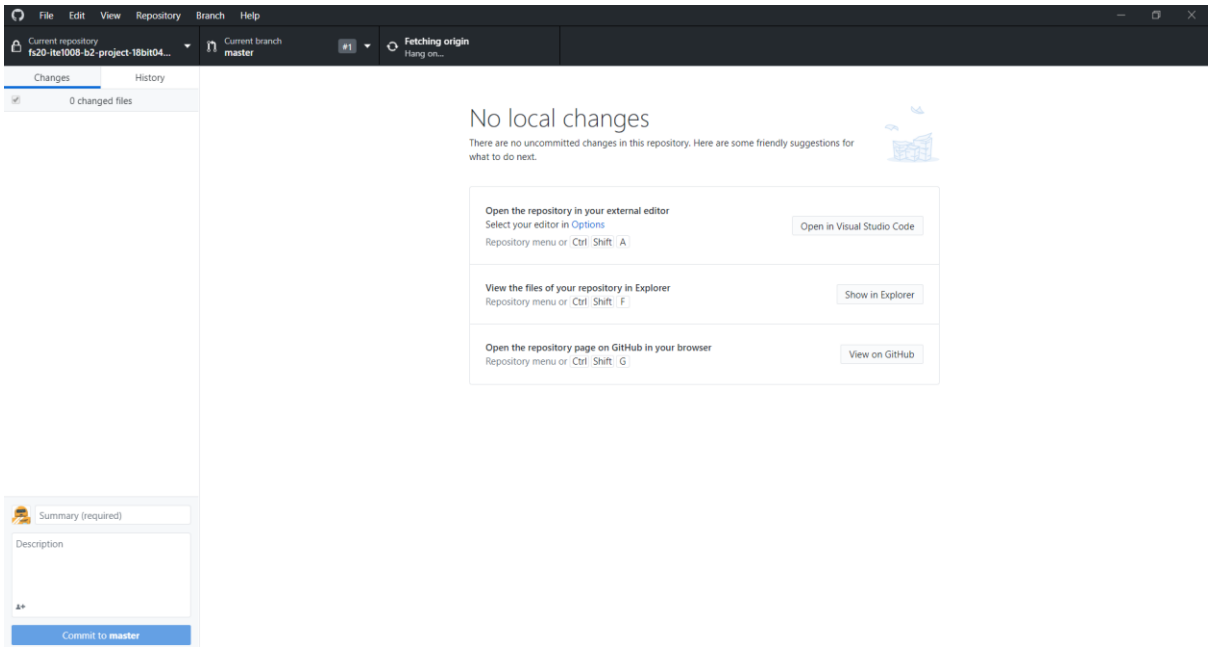
DIFFERENT WAYS TO ACCESS GITHUB

From Github Website (www.github.com) :



Most easy way to access GitHub is from the website and can work over there simple by creating the account first and then making a repository and committing changes to it.

From GitHub Desktop :

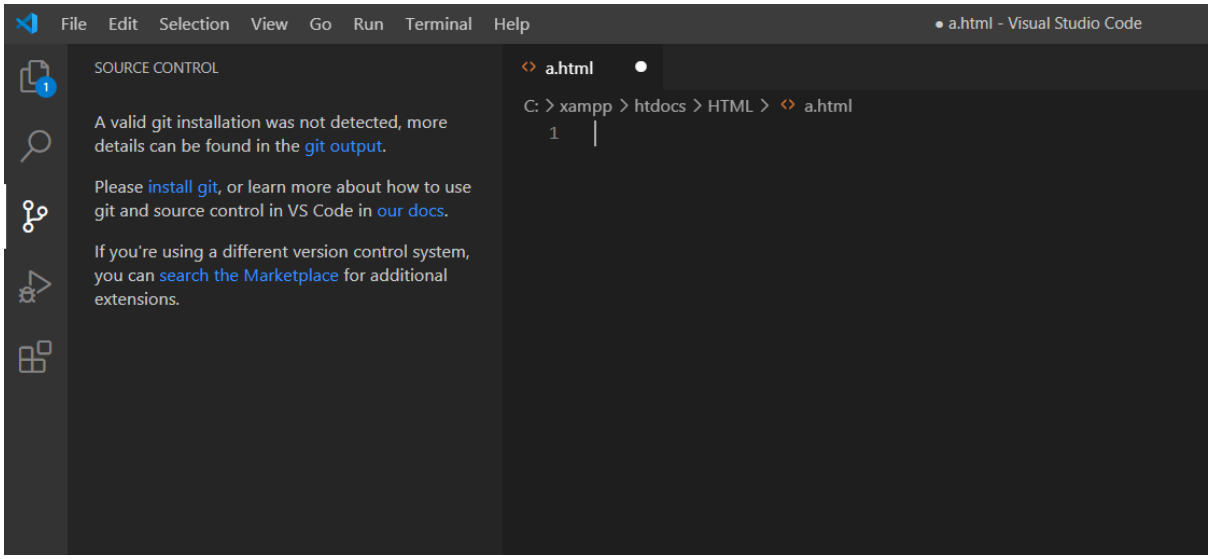


From this particular app you can access your GitHub and push and pull as well as commit to the master from here itself.

GitHub can also be accessed from VSCode :

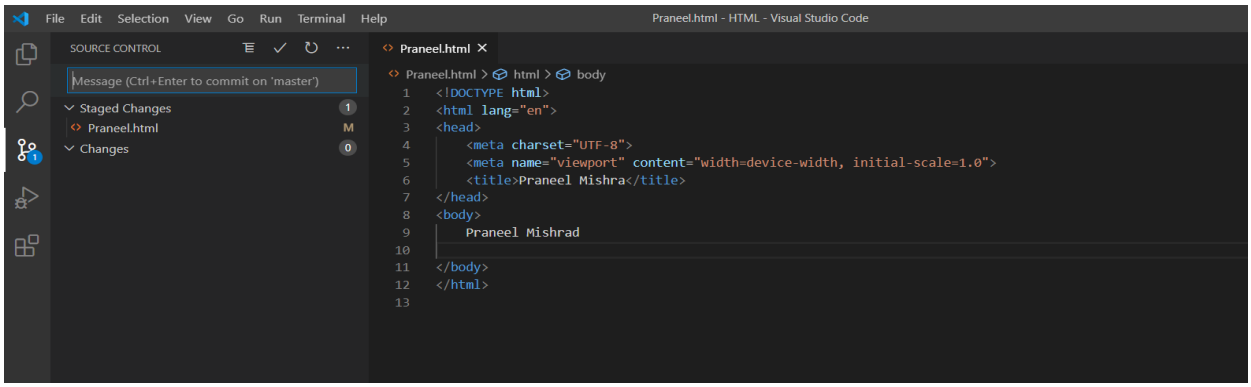
Prerequisite :

- GIT INSTALLED IN THE PC else you will get an error :

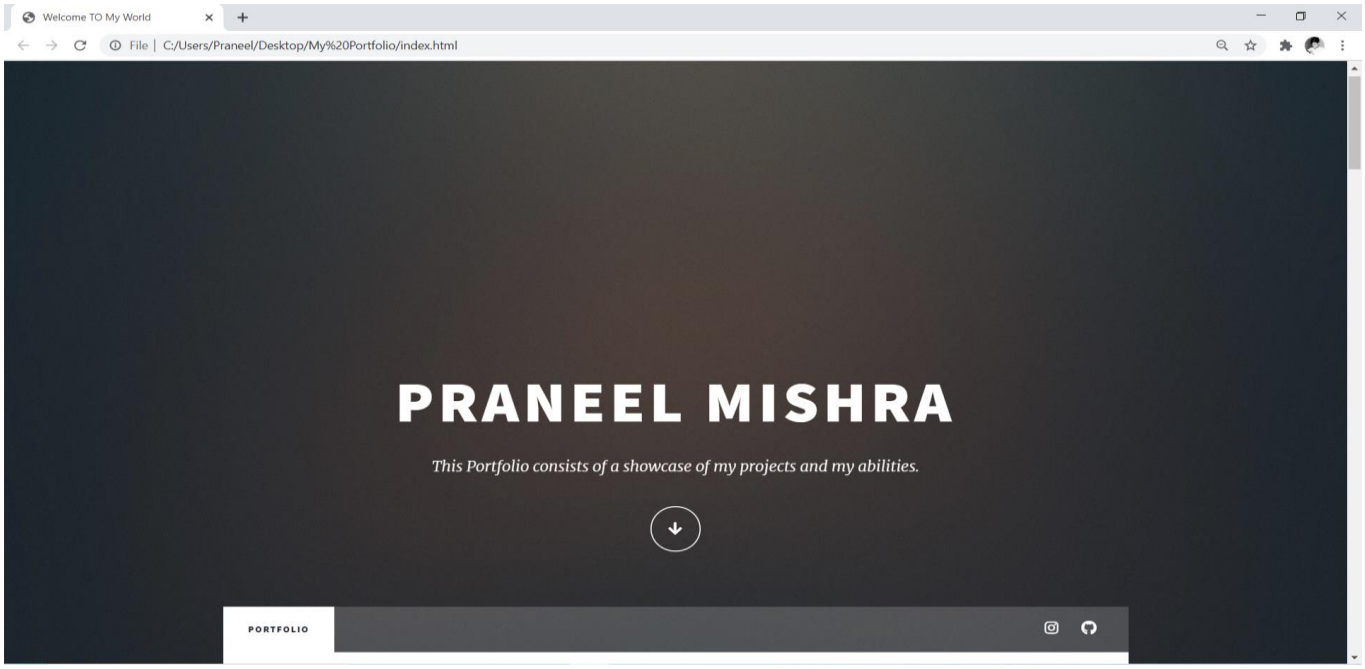


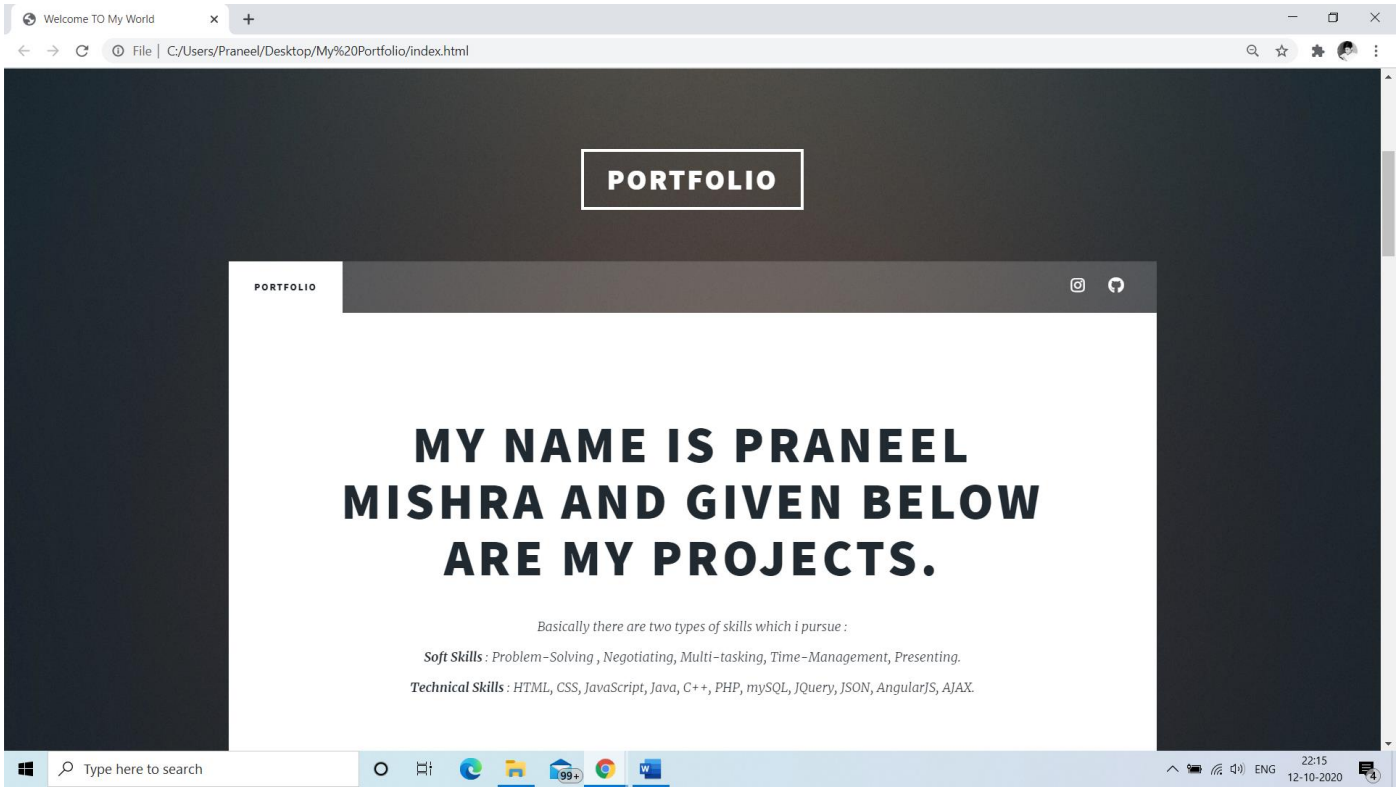
How to Access?

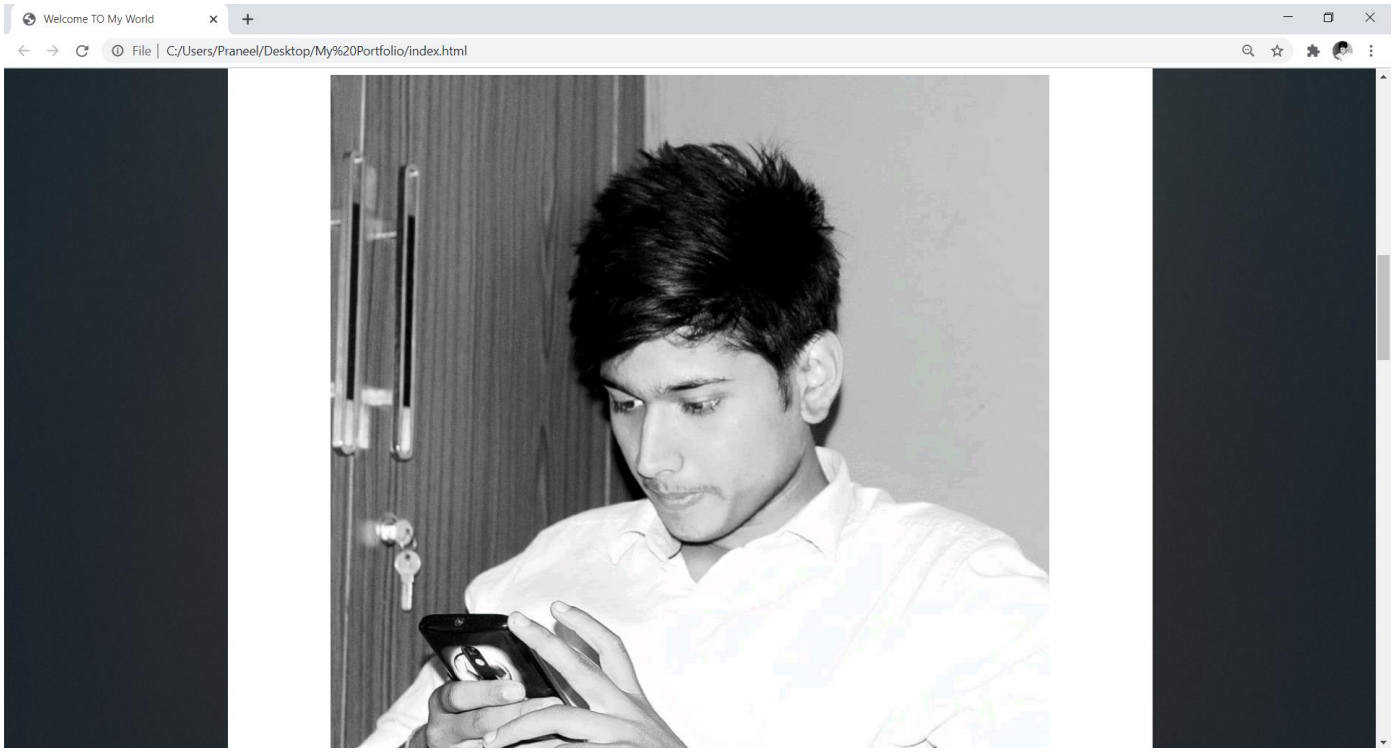
Install GIT on your PC and then clone the repository from your Github to VS Code and you will get :

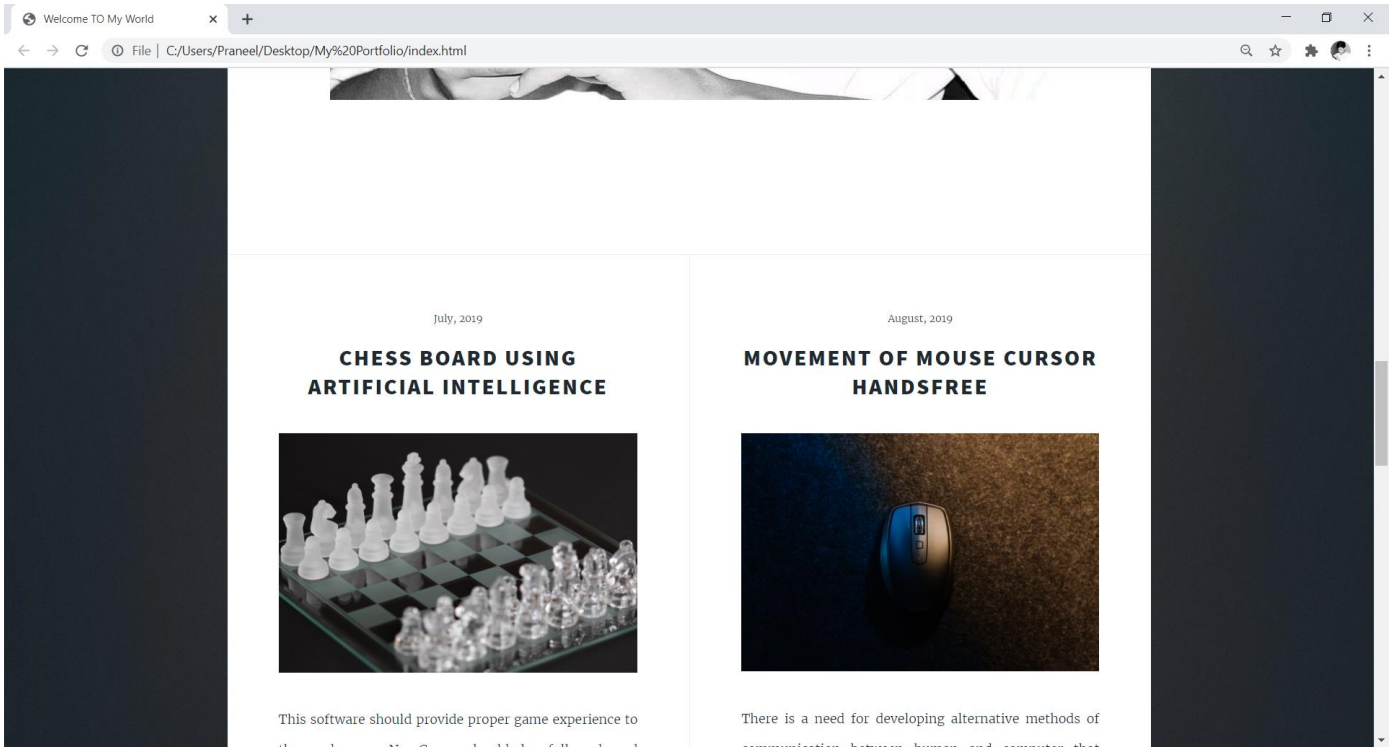


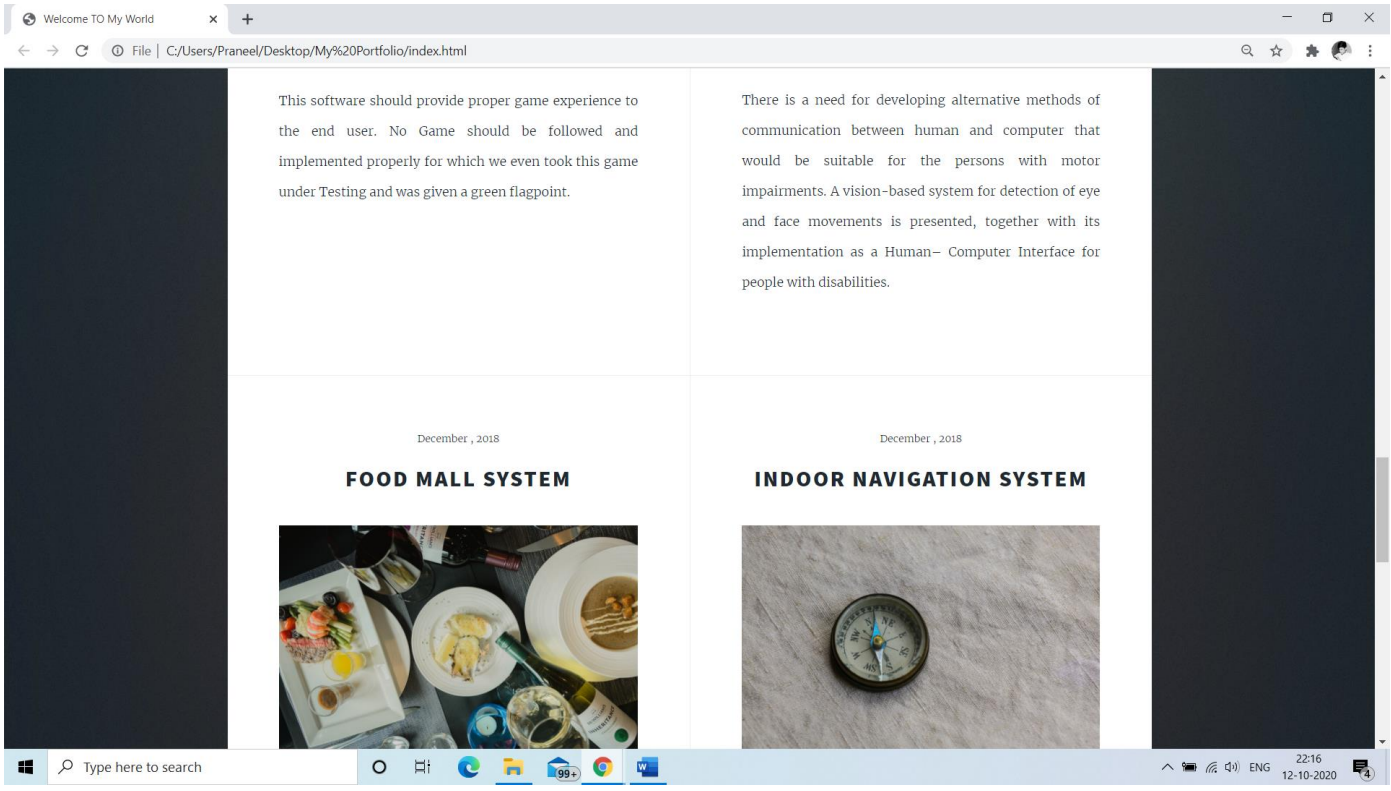
PERSONAL PORTFOLIO (SCREENSHOTS)

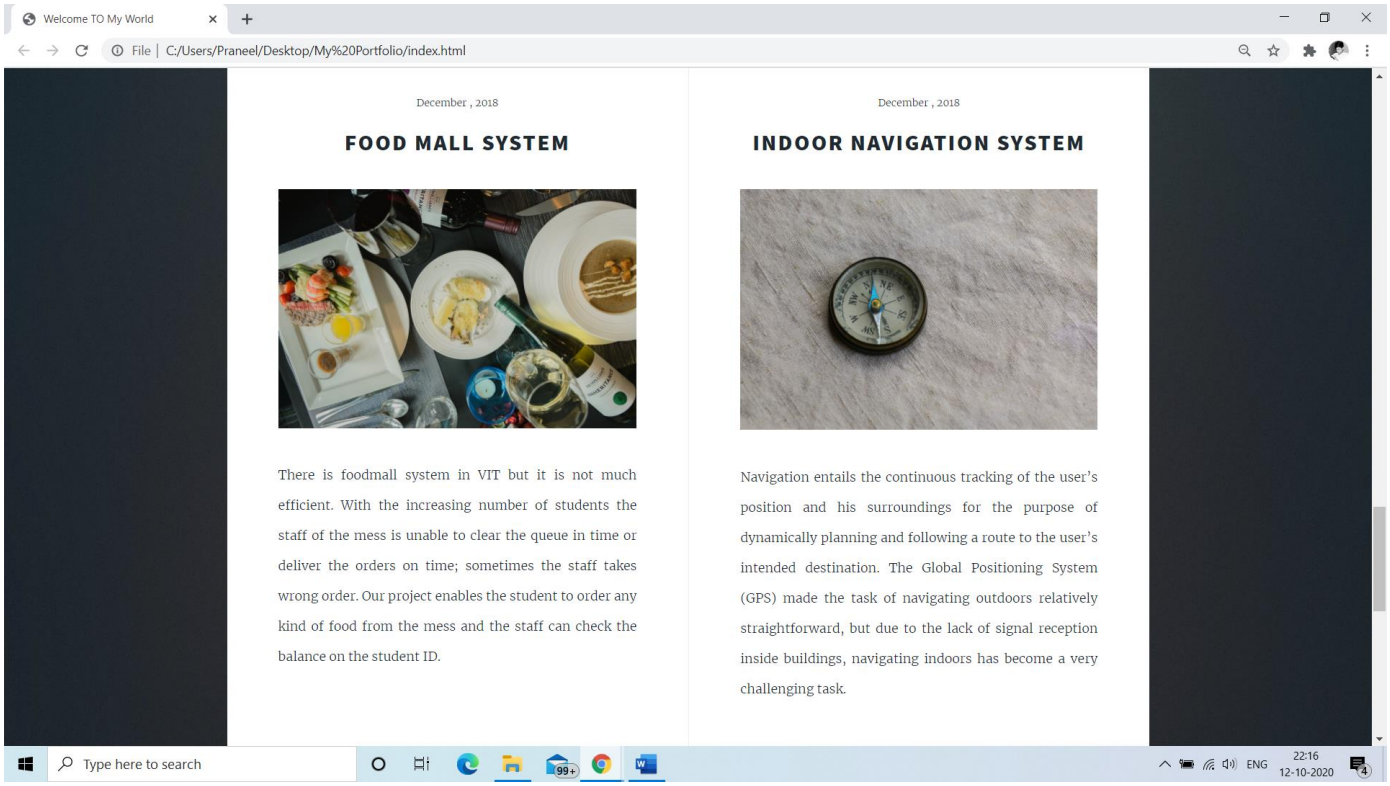


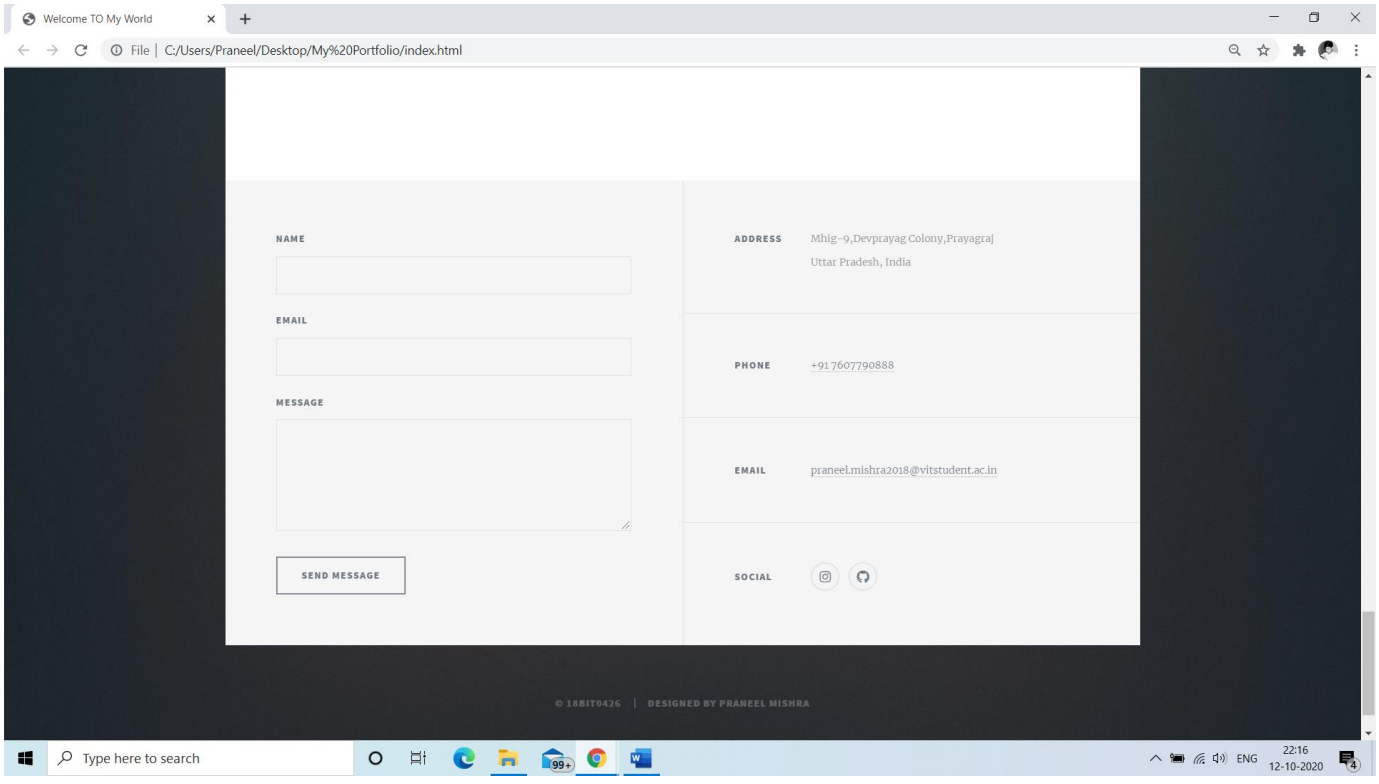

Welcome TO My World
File | C:/Users/Praneel/Desktop/My%20Portfolio/index.html
PORTFOLIO
PORTFOLIO
MY NAME IS PRANEEL MISHRA AND GIVEN BELOW ARE MY PROJECTS.
Basically there are two types of skills which i pursue :
Soft Skills : Problem-Solving , Negotiating, Multi-tasking, Time-Management, Presenting.
Technical Skills : HTML, CSS, JavaScript, Java, C++, PHP, mySQL, JQuery, JSON, AngularJS, AJAX.
Type here to search
ENG
22:15
12-10-2020


Welcome TO My World
File | C:/Users/Praneel/Desktop/My%20Portfolio/index.html


Welcome TO My World
File | C:/Users/Praneel/Desktop/My%20Portfolio/index.html
July, 2019
CHESS BOARD USING ARTIFICIAL INTELLIGENCE
This software should provide proper game experience to
August, 2019
MOVEMENT OF MOUSE CURSOR HANDSFREE
There is a need for developing alternative methods of


Welcome TO My World
File | C:/Users/Praneel/Desktop/My%20Portfolio/index.html
This software should provide proper game experience to the end user. No Game should be followed and implemented properly for which we even took this game under Testing and was given a green flagpoint.
There is a need for developing alternative methods of communication between human and computer that would be suitable for the persons with motor impairments. A vision-based system for detection of eye and face movements is presented, together with its implementation as a Human– Computer Interface for people with disabilities.
December , 2018
FOOD MALL SYSTEM
December , 2018
INDOOR NAVIGATION SYSTEM
Type here to search
ENG
22:16
12-10-2020





OTHER HTML code, CSS, JavaScript files have been added in the GitHub repository and the link is given below (google drive link is also given below) :

GOOGLE DRIVE :

- https://drive.google.com/drive/u/0/folders/1ynBYbAR4fhS_pRQaRJbZ42kSInXO22Ls

GITHUB REPOSITORY :

- https://github.com/JayakumarClassroom/fs20-ite1008-da1-b2-iampm99
- https://github.com/iampm99/PraneelMishra.github.io
- https://iampm99.github.io/PraneelMishra.github.io/

PROS AND CONS OF GITHUB

PROS :

- Multi-platform support, Linux, Windows, Ubuntu etc
- In GitHub You will see the use of simple text editor to write documents. It allows other developers to gain good knowledge by understanding simple text and gain knowledge from it.
- GitHub has a variety of users and documents hence making it a full of knowledge site by which you can learn many things from it. From making a simple project to developing good skills to work in tough situations as well and acts as a side by side knowledge provider if you get stuck at any point of your project.
- It consists of Pages as well known as GitHub Pages hence helps you to host any HTML page (like Portfolio) there itself.
- Gists in GitHub helps you to convert one or several files into a working git repository.
- GitHub helps the most in team based project if the team members are split world-wide. In GitHub every team member can commit new changes to the source code from wherever possible ,hence making it flexible for the team members through PULL and PUSH requests. It helped me during the making of project in this COVID-19 pandemic.
- GitHub creates a backup of all the files as whenever you commit any change to the repository there is a history of that particular file as well and if in future there comes an error in the file which you committed you can simple rollback to the previous file hence making it easier or the developer and also it helps to the people who wish to understand that particular project.
- While working through a project, using GitHub to view the differences between an engineer's branch and master helps the engineer work efficiently and effectively.
- Bets way to show case your skills under open source development.

CONS :

- Security is the most potential drawback of GitHub as there is no private repository for it and your repository files will be accessible worldwide.(in free cases)
- Some of the GitHub feature are only accessible when you pay for them and it is impossible or not affordable if you are a single guy working in the project, unless and until you don’t have a team you cannot go for it.
- Lack of command line configuration options and everything is GUI based.
- Reviewing large pull requests can be tedious and it can be tough to identify recent changes (e.g. a one line change) in new files or files with lots of changes.

- Knowing the difference between user and organization accounts is key, there have been many times where I've wasted minutes looking at the wrong account trying to find a relevant setting or feature. For example : There has been always a headache for me distinguishing or finding my VIT Faculty's Organization Repository.
- Organization with no paid plan cannot have private repositories.
- Migrating repositories to the services is complicated.

FEATURES NEED TO BE ADDED IN GITHUB

- Reverse the merge option (because after merging it becomes very difficult to reverse)
- When browsing history of a file, GitHub could make it easier to see the file after a particular commit instead of just being able to quickly view the commit. I'd like to be able to see the commit or the file itself in one click.
- Lacks First party control on mobile, i.e, no mobile application.
- The UI of the GitHub can be made more better and understandable for new users. (for non-technical users)
- Navigation between repositories can be very difficult if you are new to it.
- Not so good search features (Can be Improved)
- Private repositories for free accounts.
- There is also a file size restriction which should be increased as it is only 100 MB.
- GitHub Desktop can get out of sync and wonky. So it should be fixed.
- Security must be increased and to factor authentication must be added to it.
- It should be made more easy to understand as if someone is not familiar with command line then he/she can face issues working in GitHub.
- A Notification either on mail or a message on phone number is must if some changes is made to the repository.

VERSION CONTROL APPLICATIONS

Version control systems are a category of software tools that help a software team manage changes to source code over time. Version control software keeps track of every modification to the code in a special kind of database.

Types Of Version Control Systems are :

- **Git :** Git is a free and open source distributed version control system designed to handle everything from small to very large projects with speed and efficiency. **Git** is easy to learn and has a tiny footprint with lightning fast performance.
- **CVS :** CVS is a version control system, an important component of Source Configuration Management (SCM). Using it, you can record the history of sources files, and documents.
- **SVN :** Apache Subversion, also known as Subversion, SVN represents the most popular centralized version control system on the market. With a centralized system, all files and historical data are stored on a central server.
- **Mercurial :** Mercurial is a free, distributed source control management tool. It efficiently handles projects of any size and offers an easy and intuitive interface.

- **Monotone :** Monotone is a free distributed version control system. It provides a simple, single-file transactional version store, with fully disconnected operation and an efficient peer-to-peer synchronization protocol.
- **TFS :** Team Foundation Version Control (TFVC) is a centralized version control system. Typically, team members have only one version of each file on their dev machines.

Comparison between these and GIT is given below

GIT vs CVS vs SVN

- CVS is ancient in software terms. Its initial release is now 26 years old, and the latest release is already 9 years old. Its maintenance has stopped, so it is not advised to use it anymore. However, it was really good in a lot of things, even though it tracked only files, instead of a whole project (although you may argue here). It was a bit problematic to share your work with others, until pserver came to existence, but it was not trivial to use (I may be biased here).

- Subversion (SVN) superseded CVS in many ways. It was initially released 16 years ago, and its latest release is only 6 months old. It is still actively maintained and developed by the Apache Foundation. Originally, it was centralised by nature, which means you had to have a SVN server to even store your changes. Sharing work with others required those others to have access to the same server. The SVN developers now work on an "offline mode", which means you can stare your changes locally, but you will still have to upload them to the server to share it with others (citation needed). There are some online services available that offer SVN hosting, probably the most notable being Assembla.

- Git was initially released 12 years ago, and was created by Linus Torvalds, the creator of Linux. Its latest release is only a few days old. It is actively maintained and developed, thanks to the hype created by GitHub. It is decentralised by nature, which means you don't need a Git server to store your changes, even though you can have one. It is easy to share your work with others, even as a whole, or partially (ie. commit per commit.) There are more and more service providers out there who help you host your Git repositories, the most notable ones being GitHub, GitLab, and Bitbucket.

In the end it all comes own under the requirements.

If you need a strongly centralised environment where developers should only push their changes to golden repository you should probably use SVN (or TFS, or Perforce, etc.) If you want to give more freedom to your developers, ie. letting them share change sets between each other, you may want to use Git (or Mercurial, or Fossil, etc.) instead.

GITHUB LINK

- **https://github.com/JayakumarClassroom/fs20-ite1008-da1-b2-iampm99**
- **https://github.com/iampm99/PraneelMishra.github.io**
- **https://iampm99.github.io/PraneelMishra.github.io/**