

Submitted in partial fulfillment of the requirements

of the degree of

**B. E. Computer Engineering**

**A.Y (2019-2020)**

**FaceGram on Heroku**

By

**Vidhi Doshi 60004160023**

**Ruturaj Gujar 60004160035**

**Aayush Jain 60004160036**

**Guide(s):**

**Sonali Jadhav**

**Chandana Nighut**

**ACKNOWLEDGEMENT**

We would like to express our sincere gratitude to our guides, Prof.Sonali Jadhav

Chandana Nighut, for their guidance, encouragement and gracious support throughout the course of our work, for their expertise in the field motivated us to work in this area and for their faith in us in every stage of research.

We are grateful to Dr. Meera Narvekar, Head of the Department of Computer Engineering, for

letting us use the department resources, labs, and books.

We are highly indebted to our Principal, Dr Hari Vasudevan, for availing us with proper and

resourceful journals like IEEE, ScienceDirect and other online material that helped us finish our

project with ease and perfection.

We would also like to thank all our fellow students and the staff of the Department of Computer

Engineering for their help in the process leading to the conceptualization of this project.

**TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

**ABSTRACT**

Social Networking describes the phenomena found in, participatory and self-expressive Web sites—such as YouTube, MySpace, Facebook—where members/participants expose, discuss, reveal, and expound on their personal lives, activities, hopes, dreams for others to see and marvel upon.

This project aims to explore the cloud application Heroku by hosting FaceGram (social networking website) on it. Heroku is a cloud platform as a service (PaaS) supporting several programming languages and is based on a managed container system, with integrated data services and a powerful ecosystem, for deploying and running modern apps.

In this project, we are going to analyze different features of Heroku such as continuous integration, elastic runtime and how it easily scales apps in a single click with no downtime.

**INTRODUCTION**

Social networking sites allow users to share ideas, digital photos and videos, posts, and to inform others about online or real-world activities and events with people in their network. While in-person social networking – such as gathering in a village market to talk about events – has existed since the earliest development of towns, the web enables people to connect with others who live in different locations, ranging from across a city to across the world.

FaceGram is a web app that inherits the best features from multiple social networking sites and combines them into a single site. The features included in our app are:

1. Posting Text Statuses:   
   The users can post any text status on the app.
2. Posting Images:  
   Along with text status, the users are also allowed to post images.
3. Liking and Commenting on the Posts:  
   Other users can like or comment on their friend’s posts.
4. Finding friends and sending them a request:  
   Users can find friends through the search bar and send them a request. If the request is accepted, the friend is added to the user’s friend list.
5. Chatting with friends:

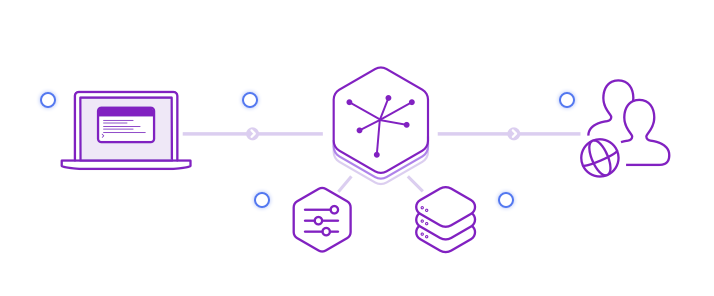
User can chat with their friends anytime.

1. Profile page:  
   This is a page where users can view their bio, past posts, etc. Users can also view a friend’s profile page.
2. Notifications:

Notifications to update user about any recent event. For eg. If a user gets a friend request then it is shown in the notifications.

The webapp is hosted on heroku and is now fully functional and always online.

**LITERATURE SURVEY**

****

**HEROKU MODEL**

Heroku is a platform as a service based on a managed container system, with integrated data services and a powerful ecosystem, for deploying and running modern apps. The Heroku developer experience is an app-centric approach for software delivery, integrated with today’s most popular developer tools and workflows.

In the above figure,the developer first deploys the code.The app is then operated through smart containers or Dynos.The developer manages app through dashboard and data is stored in web-backing services (via add-ons).Users then make requests which are served by the app.

**Heroku Runtime**

Heroku runs your apps inside dynos — smart containers on a reliable, fully managed runtime environment. Developers deploy their code written in Node, Ruby, Java, PHP, Python, Go, Scala, or Clojure to a build system which produces an app that's ready for execution. The system and language stacks are monitored, patched, and upgraded, so it's always ready and up-to-date. The runtime keeps apps running without any manual intervention.

**Heroku Developer Experience (DX)**

The Heroku Developer Experience is an app-centric approach to software delivery so developers can focus on creating and continuously delivering applications, without being distracted by servers or infrastructure. Developers deploy directly from popular tools like Git, GitHub or Continuous Integration (CI) systems. The intuitive web-based Heroku Dashboard makes it easy to manage your app and gain greater visibility into performance.

**Data Services and Ecosystem**

Heroku Elements let developers extend their apps with Add-ons, customize their application stack with Buildpacks and jumpstart their projects with Buttons. Add-ons are 3rd party cloud services that developers can use to immediately extend their apps with a range of functionality such as data stores, logging, monitoring and more. Heroku provides three fully-managed data service Add-ons: Heroku Postgres, Heroku Redis, and Apache Kafka on Heroku.

**Heroku Operational Experience (OpEx)**

The Heroku Operational Experience is a key component of the platform. It helps developers through troubleshooting and remediation of common issues and customizing their ops experience to quickly identify and address negative trends in their application health. Heroku provides a set of tools to alert you if something goes wrong, or to automatically scale your web dynos if the response time for web requests exceeds a threshold you specify. Application metrics, Threshold Alerting, and Autoscaling are some of the features you get access to with no extra cost.

**Security and Compliance**

Developers from around the world entrust sensitive data to Heroku, and nothing is more important to us than honoring our custodial commitments to protect this data. Heroku regularly performs audits and maintains PCI, HIPAA, ISO, and SOC compliance to further strengthen our trust with customers. Learn more by visiting our compliance center.

The concepts can be divided into two buckets: those that involve the development and deployment of an application, and those that involve the runtime operation of the Heroku platform and the application after it’s deployed.

The following two sections explain the main components of the platform, separating them into these two buckets.

### [**Deploy**](https://devcenter.heroku.com/articles/how-heroku-works#deploy)

* Applications consist of your source code, a description of any dependencies, and a Procfile.
* [Procfiles](https://devcenter.heroku.com/articles/procfile) list process types - named commands that you may want executed.
* Deploying applications involves sending the application to Heroku using either Git, GitHub, or via an API.
* [Buildpacks](https://devcenter.heroku.com/articles/buildpacks) lie behind the slug compilation process. Buildpacks take your application, its dependencies, and the language runtime, and produce slugs.
* A [slug](https://devcenter.heroku.com/articles/slug-compiler) is a bundle of your source, fetched dependencies, the language runtime, and compiled/generated output of the build system - ready for execution.
* [Config vars](https://devcenter.heroku.com/articles/config-vars) contain customizable configuration data that can be changed independently of your source code. The configuration is exposed to a running application via environment variables.
* [Add-ons](https://elements.heroku.com/addons/) are third party, specialized, value-added cloud services that can be easily attached to an application, extending its functionality.
* A [release](https://devcenter.heroku.com/articles/releases) is a combination of a slug (your application), config vars and add-ons. Heroku maintains an append-only ledger of releases you make.

### [**Runtime**](https://devcenter.heroku.com/articles/how-heroku-works#runtime)

* [Dynos](https://devcenter.heroku.com/articles/dynos) are isolated, virtualized Unix containers, that provide the environment required to run an application.
* Your application’s [dyno formation](https://devcenter.heroku.com/articles/scaling#dyno-formation) is the total number of currently-executing dynos, divided between the various process types you have scaled.
* The [dyno manager](https://devcenter.heroku.com/articles/dynos) is responsible for managing dynos across all applications running on Heroku.
* Applications that use the free dyno type will [sleep](https://devcenter.heroku.com/articles/free-dyno-hours) after 30 minutes of inactivity. Scaling to multiple web dynos, or a different dyno type, will avoid this.
* [One-off Dynos](https://devcenter.heroku.com/articles/one-off-dynos) are temporary dynos that run with their input/output attached to your local terminal. They’re loaded with your latest release.
* Each dyno gets its own [ephemeral filesystem](https://devcenter.heroku.com/articles/dynos#ephemeral-filesystem) - with a fresh copy of the most recent release. It can be used as temporary scratchpad, but changes to the filesystem are not reflected to other dynos.
* [Logplex](https://devcenter.heroku.com/articles/logplex) automatically collates log entries from all the running dynos of your app, as well as other components such as the routers, providing a single source of activity.
* [Scaling](https://devcenter.heroku.com/articles/scaling) an application involves varying the number of dynos of each process type.

**PROBLEM DEFINITION**

Our main task in this project was to host a web app on a remote server and use cloud services to keep it online. Web hosting is a service that allows organizations and individuals to post a website or web page onto the Internet. A web host, or web hosting service provider, is a business that provides the technologies and services needed for the website or webpage to be viewed in the Internet. Websites are hosted, or stored, on special computers called servers. When Internet users want to view your website, all they need to do is type your website address or domain into their browser. Their computer will then connect to your server and your webpages will be delivered to them through the browser.

Most hosting companies require that you own your domain in order to host with them. If you do not have a domain, the hosting companies will help you purchase one.



**Fig:** **Web Hosting Service**

We will be making use of Heroku which is a type of PaaS (Platform as a Service). Platform as a service (PaaS) is a complete development and deployment environment in the cloud, with resources that enable you to deliver everything from simple cloud-based apps to sophisticated, cloud-enabled enterprise applications. You purchase the resources you need from a cloud service provider on a pay-as-you-go basis and access them over a secure Internet connection.

Like IaaS, PaaS includes infrastructure—servers, storage and networking—but also middleware, development tools, business intelligence (BI) services, database management systems and more. PaaS is designed to support the complete web application lifecycle: building, testing, deploying, managing and updating.

**METHODOLOGY**

Heroku is an open platform. There are no proprietary hooks or dependencies required to deploy and run your application.Heroku is an embodiment of application development principles and differs from traditional server-based hosting in the following ways:

* it is application, not infrastructure, focused
* is a dynamic and distributed runtime environment
* utilizes a process-based execution model
* enforces a strict separation of apps and their dependencies

On Heroku, these differences are manifested in a variety of practices, all of which result in a well-designed and resilient application and are not platform-specific.

**Step 1 : Open the command prompt.**

Navigate to the directory where your Laravel project is located. Login to your Heroku account by running : ***heroku login***

**Step 2:Run the following commands**

***git init*** *//to initialize a new repo*

***git add --all***

***git commit -m "my first commit"***

**Step 3 : Create a Procfile**

Heroku launches an Apache server with PHP to serve the app from the root of the project which in our case is the public/ folder. A Procfile is required to configure it.

Create a file at the root of your project and name it Procfile.

***echo "web: vendor/bin/heroku-php-apache2 public/" > Procfile***

***git add Procfile***

***git commit -m "Heroku Procfile"***

**Step 4 : Create Heroku App**

***heroku create <myappname>***

**Step 5 : Specify Language**

To create a buildpack that will specify PHP as the app language run : ***set heroku/php***

**Step 6 : Generate App Key for Laravel App**

Run php artisan key:***generate --show***

Copy the generated key and run the command below setting the APP\_KEY value to the key you copied.

***heroku config:set APP\_KEY = <key generated above>***

**Step 6 : Host the App on Heroku**

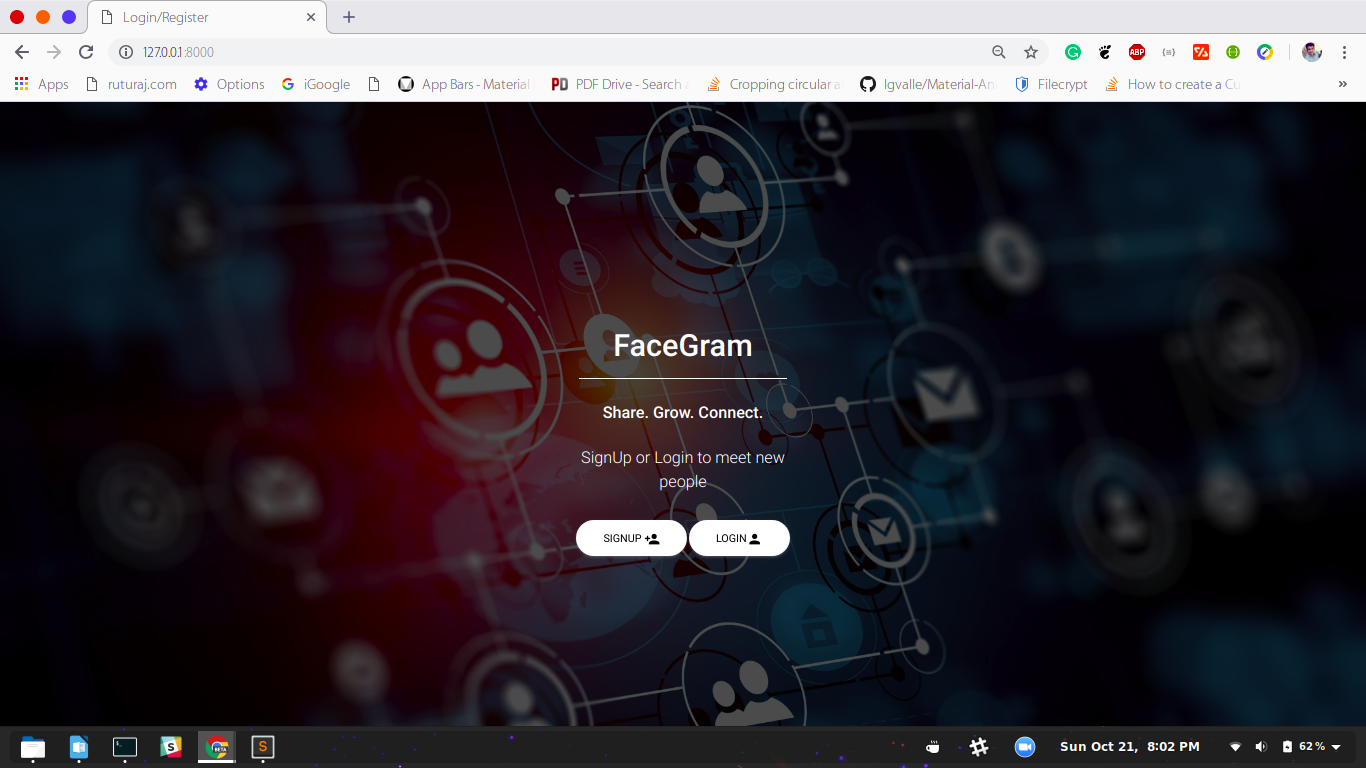
**git push heroku master --app <myappname>**

To launch the app from command line run

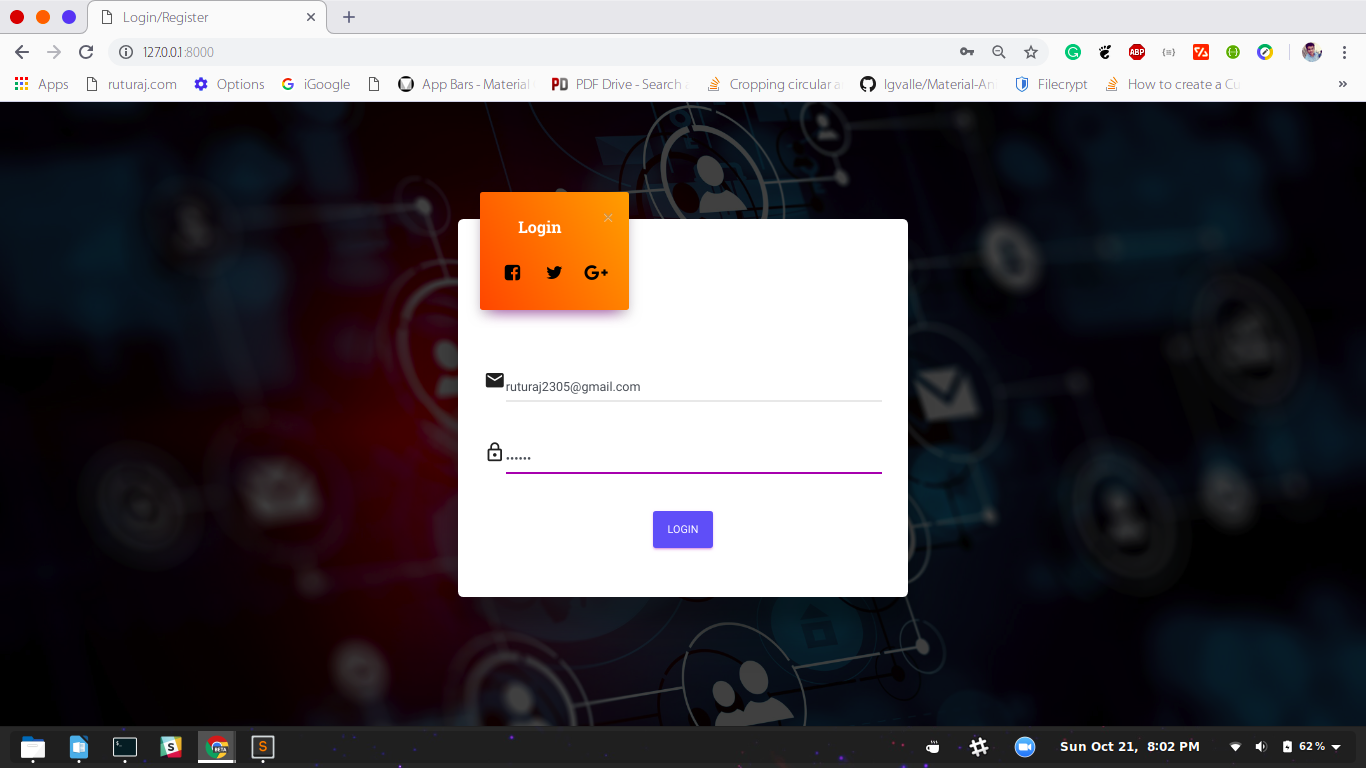
**heroku open --app <myappname>**

**RESULT ANALYSIS**

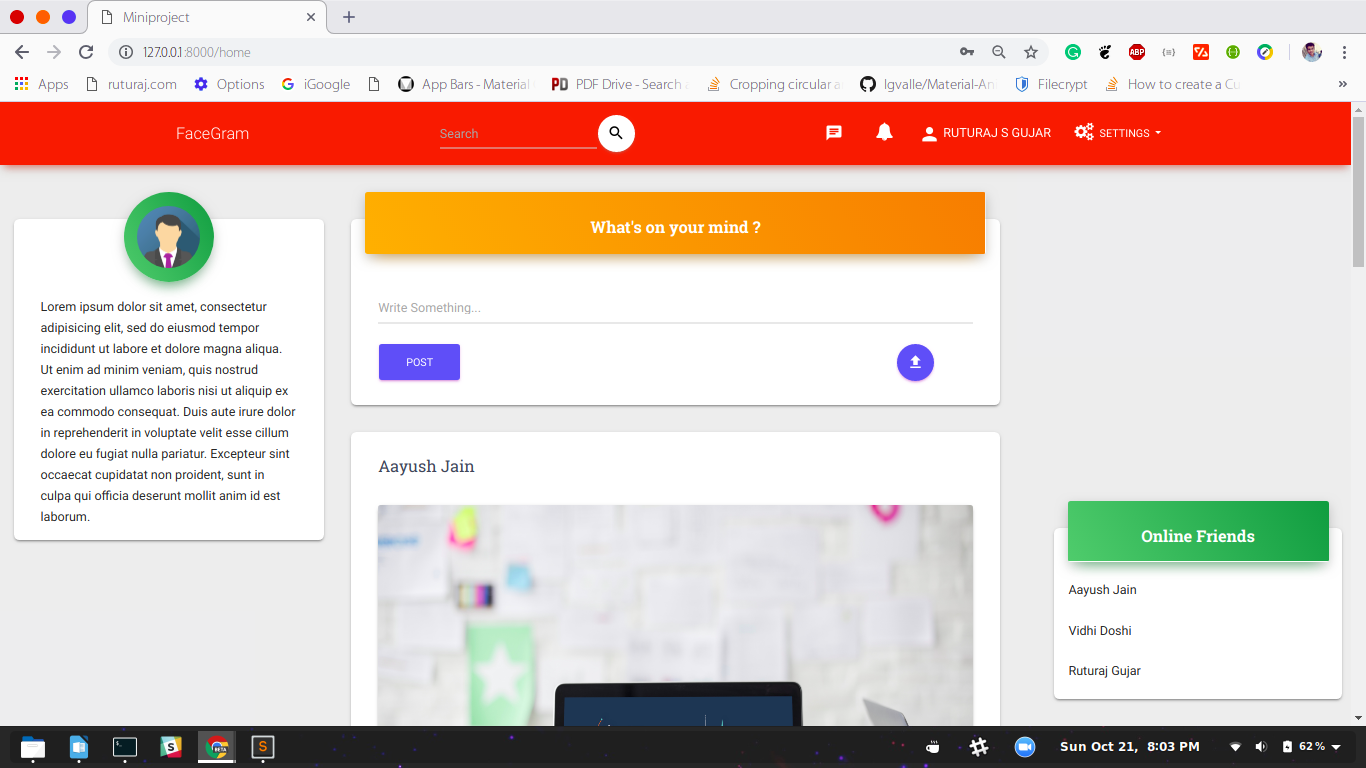
1. **Landing Page**

****

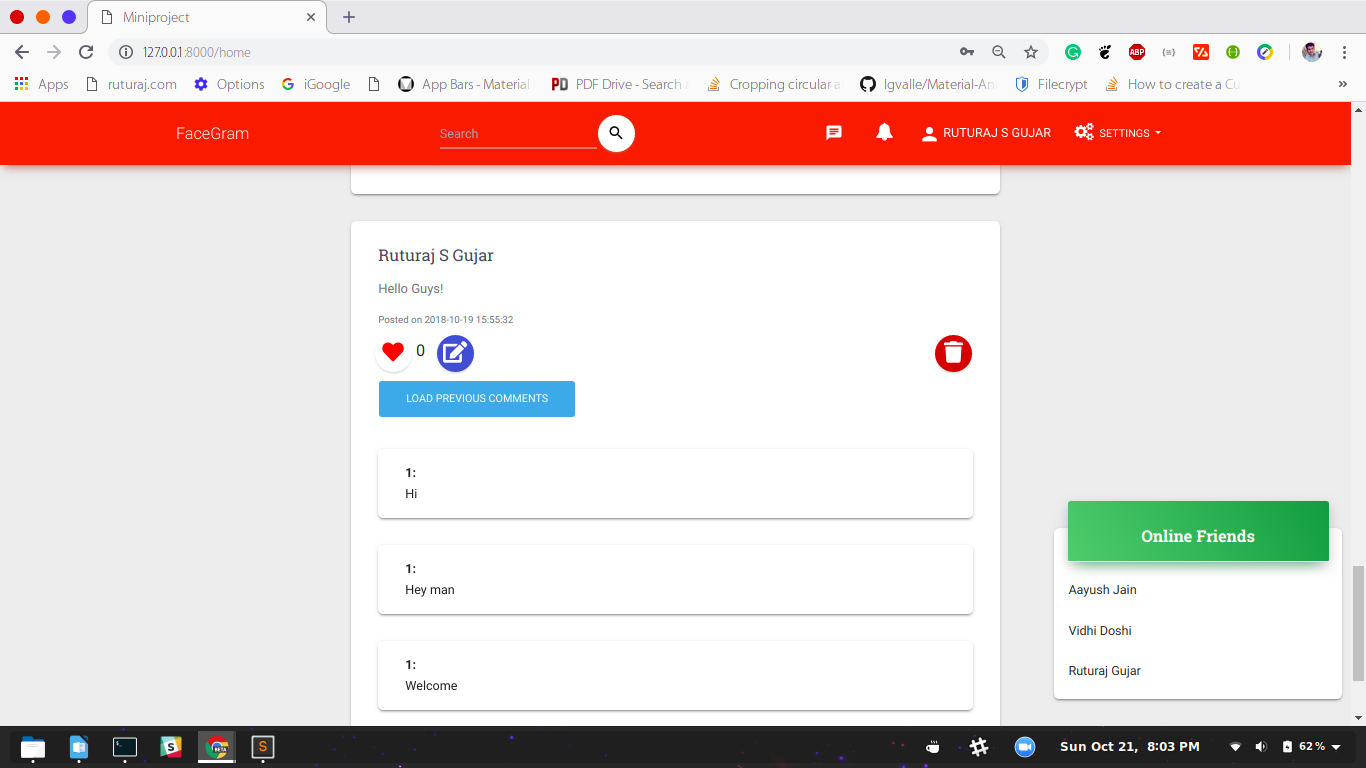
1. **Register/Login**

****

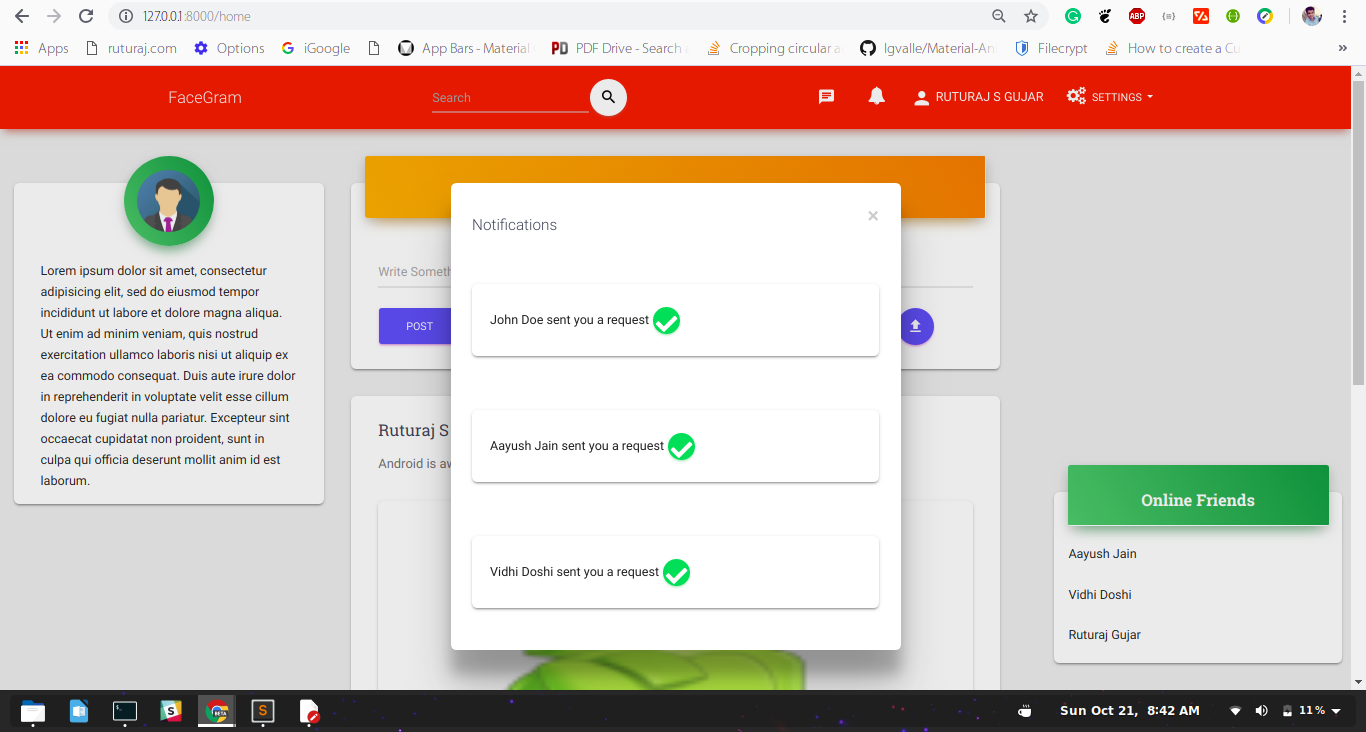
1. **Home Page**

****

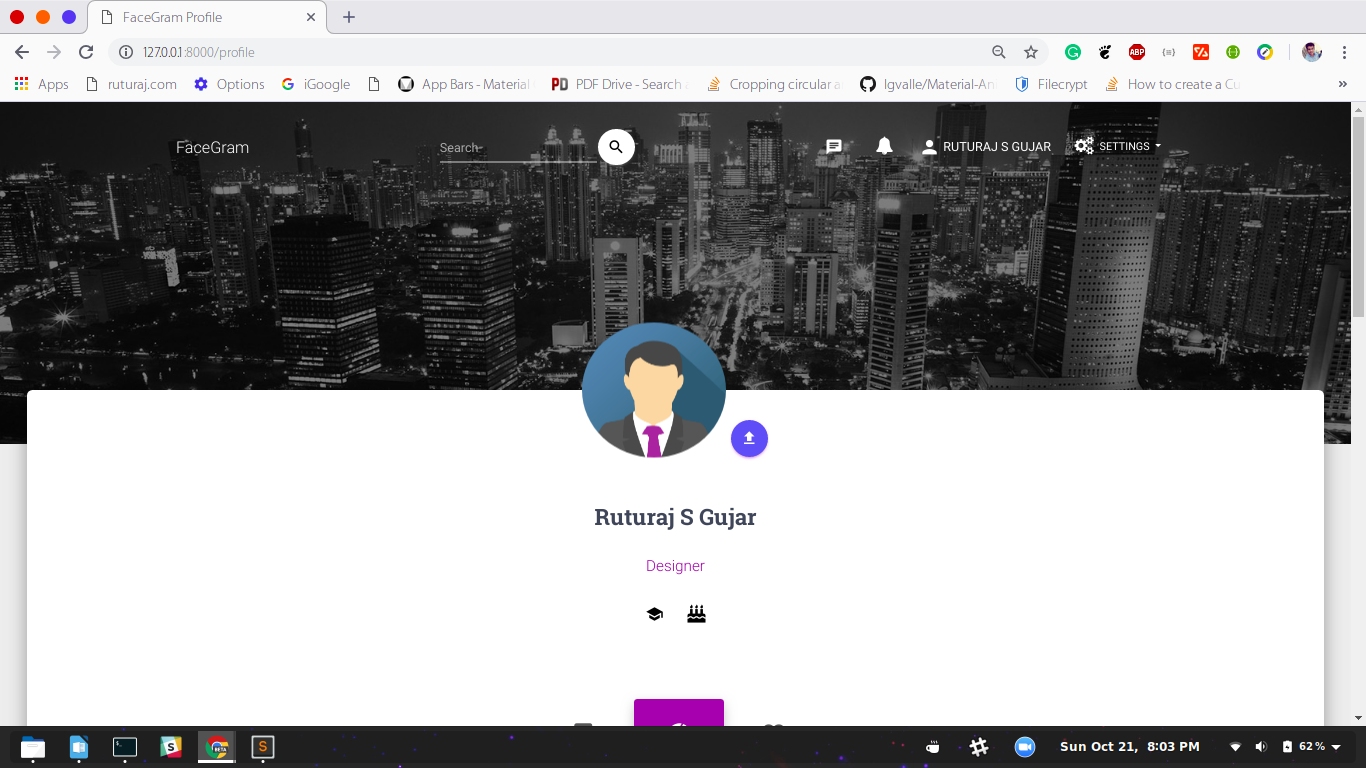
1. **Posts**

****

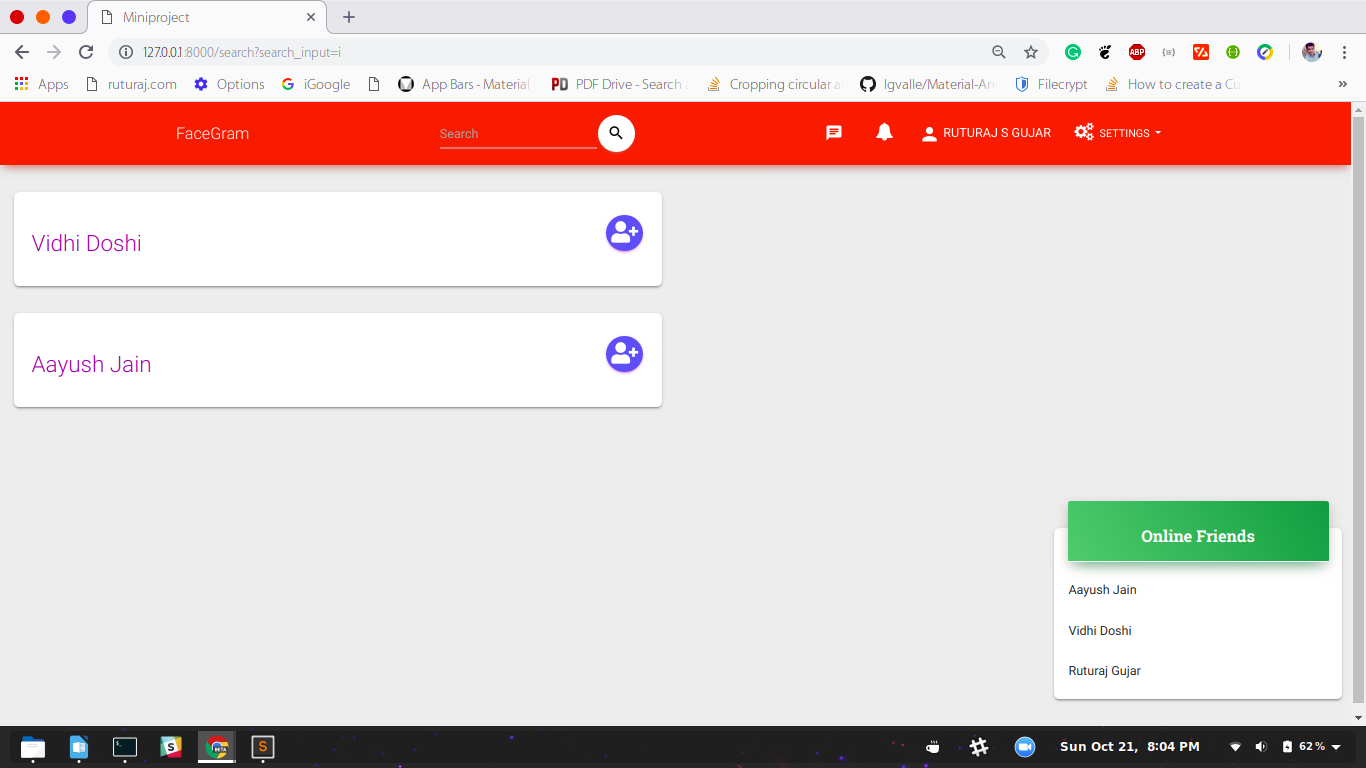
1. **Notifications**

****

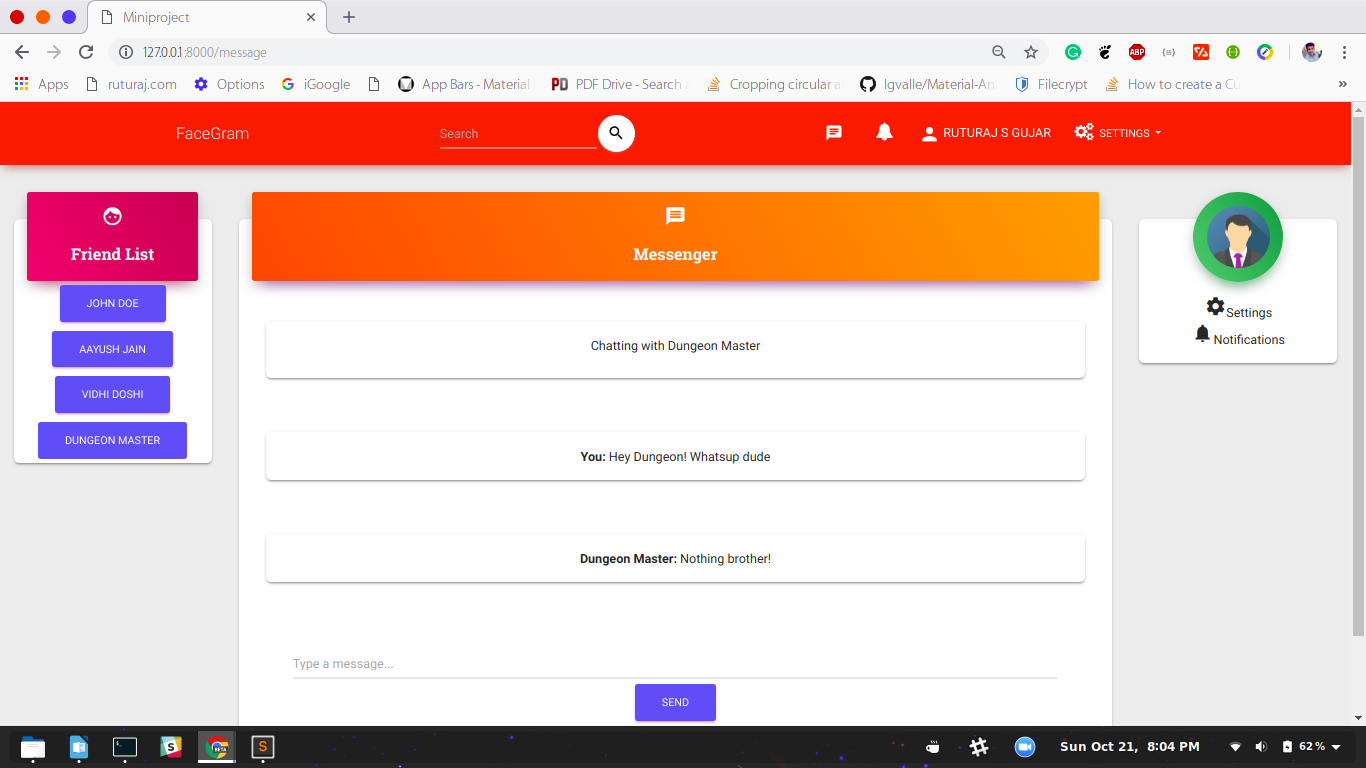
1. **Profile Page**

****

1. **Find Friends**

****

1. **Messenger**

****

**CONCLUSION**

Thus we have successfully implemented the Social Networking Site and hosted it on the cloud. In the process, we also learned about PaaS and Hosting on the web.

**FUTURE SCOPE**

Application deployment is changing. In relatively short order I’ve gone from buying hardware, to monthly hosting, to metered CPU time, and from building my open-source software manually, to package managers, to fancy config tools and recipes to pre-build whole machine images.When technology breakthroughs make something smaller, or faster, or cheaper, it doesn’t just change current use; it creates whole new type of useThis coming decade is going to see an agile revolution for the *deployment* side of the equation. The manual, guesswork-heavy methods of provisioning that we use today are soon to be superseded by methods that will make deploying an app fast, easy, and fun.

**REFERENCES**