

## **EXPERIMENT NO. 2**

**AIM:** To understand Version Control System / Source Code Management, install git and create a GitHub account.

### **THEORY:**

#### **What Is Version Control?**

A system called version control, sometimes referred to as source control or revision control, keeps track of changes made to a file or group of files over time so that you may retrieve particular versions at a later time. Although it can be applied to any circumstance where several versions of something are made and may need to be monitored and recalled, it is most frequently employed in software development.

#### **What is GIT?**

- a. 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.
- b. Git relies on the basis of distributed development of software where more than one developer may have access to the source code of a specific application and can modify changes to it that may be seen by other developers.
- c. Initially designed and developed by Linus Torvalds for Linux kernel development in 2005.
- d. Every git working directory is a full-fledged repository with complete history and full version tracking capabilities, independent of network access or a central server.
- e. Git allows a team of people to work together, all using the same files. It helps the team cope with the confusion that tends to happen when multiple people are editing the same files.

#### **What Is GitHub?**

GitHub is a web-based platform which hosts software development projects and uses Git for version management. Git is a distributed version control system that helps developers to work together on same software projects and keep track of changes made to their code by on another. GitHub offers a user-friendly interface, which is very collaborative tools, and more project management tools, GitHub will enhance the potential of the Git.

GitHub allows developers to create and manage the code in the repository in the remote location where others can access the code or GitHub is a collection repository which contains the files of the project.

#### **What is Use of Version Control Software?**

- a. Version control software allows the user to have “versions” of a project, which show the changes that were made to the code over time, and allows the user to backtrack if necessary and undo those changes.
- b. This ability alone – of being able to compare two versions or reverse changes, makes it fairly invaluable when working on larger projects.

- c. In a version control system, the changes would be saved just in time – a patch file that could be applied to one version, in order to make it the same as the next version.
- d. All versions are stored on a central server, and individual developers' checkout and upload changes back to this server.

### **What Are the Uses Cases of GitHub?**

Following is some of the use cases of GitHub:

- a. Version Control: GitHub is also called a version control system because of it uses such as if the certain developers are working on the same project and if any developer make changes at it is affecting the entire code, then they move back to previous version with immediate actions.
- b. Collaboration and Code Review: GitHub allows group of developers work on same project where there can review the each other's code and can work on the same project which will improve the productivity and where they can develop the complex application in faster manner.
- c. Issue Tracking: Has a certain group of developers will work on the same project so when the issue arises in the GitHub then you can assign the issue to the other developer to whom you want.
- d. Open-Source Development: The most widely used platform for open-source development is GitHub.

### **What Are Characteristics of Git?**

#### ***A. Strong support for non-linear development***

- a. Git supports rapid branching and merging and includes specific tools for visualizing and navigating a non-linear development history.
- b. A major assumption in Git is that a change will be merged more often than it is written.
- c. Branches in Git are very lightweight.

#### ***B. Distributed development***

- a. Git provides each developer a local copy of the entire development history, and changes are copied from one such repository to another.
- b. The changes can be merged in the same way as a locally developed branch very efficiently and effectively.

#### ***C. Compatibility with existing systems/protocol***

- a. Git has a CVS server emulation, which enables the use of existing CVS clients and IDE plugins to access Git repositories.

#### ***D. Efficient handling of large projects***

- a. Git is very fast and scalable compared to other version control systems.
- b. The fetching power from a local repository is much faster than is possible with a remote server.

### ***E. Data Assurance***

- a. The Git history is stored in such a way that the ID of a particular version depends upon the complete development history leading up to that commit.
- b. Once published, it is not possible to change the old versions without them being noticed.

### ***F. Automatic Garbage Collection***

- a. Git automatically performs garbage collection when enough loose objects have been created in the repository.
- b. Garbage collection can be called explicitly using `git gc --prune`.

### ***G. Periodic explicit object packing***

- a. Git stores each newly created object as a separate file. It uses packs that store a large number of objects in a single file (or network byte stream) called pack file, delta-compressed among themselves.
- b. A corresponding index file is created for each pack file, specifying the offset of each object in the pack file.
- c. The process of packing can be very expensive computationally.
- d. Git allows the expensive pack operation to be deferred until later when time does not matter.
- e. Git does periodic repacking automatically but manual repacking can be done with the `git gc` command.

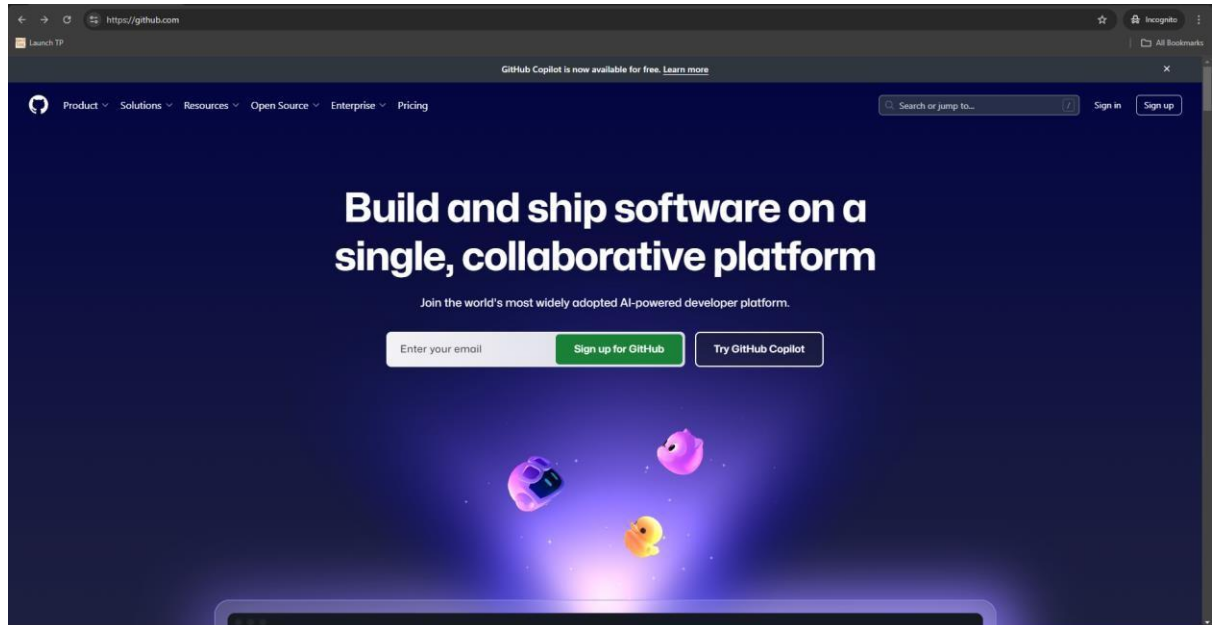
### **How does GIT work?**

- a. A Git repository is a key-value object store where all objects are indexed by their SHA-1 hash value.
- b. All commits, files, tags, and filesystem tree nodes are different types of objects living in this repository.
- c. A Git repository is a large hash table with no provision made for hash collisions.
- d. Git specifically works by taking “snapshots” of files

Heramb Vengurlekar  
T23 – 120

## Steps to creating a GitHub account:

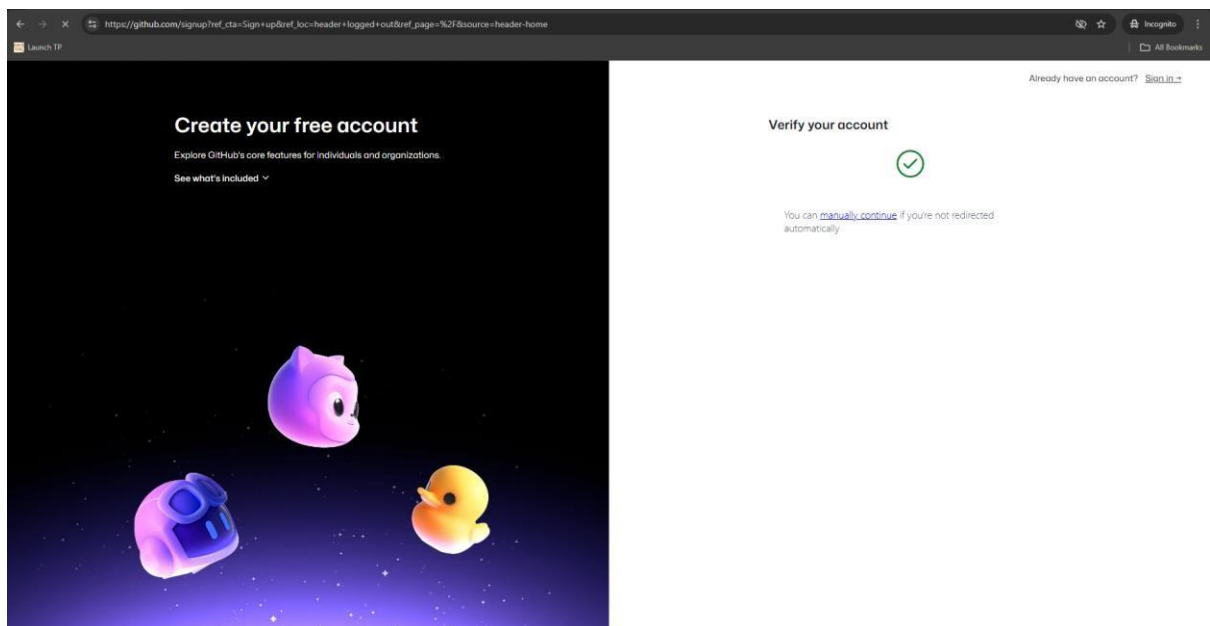
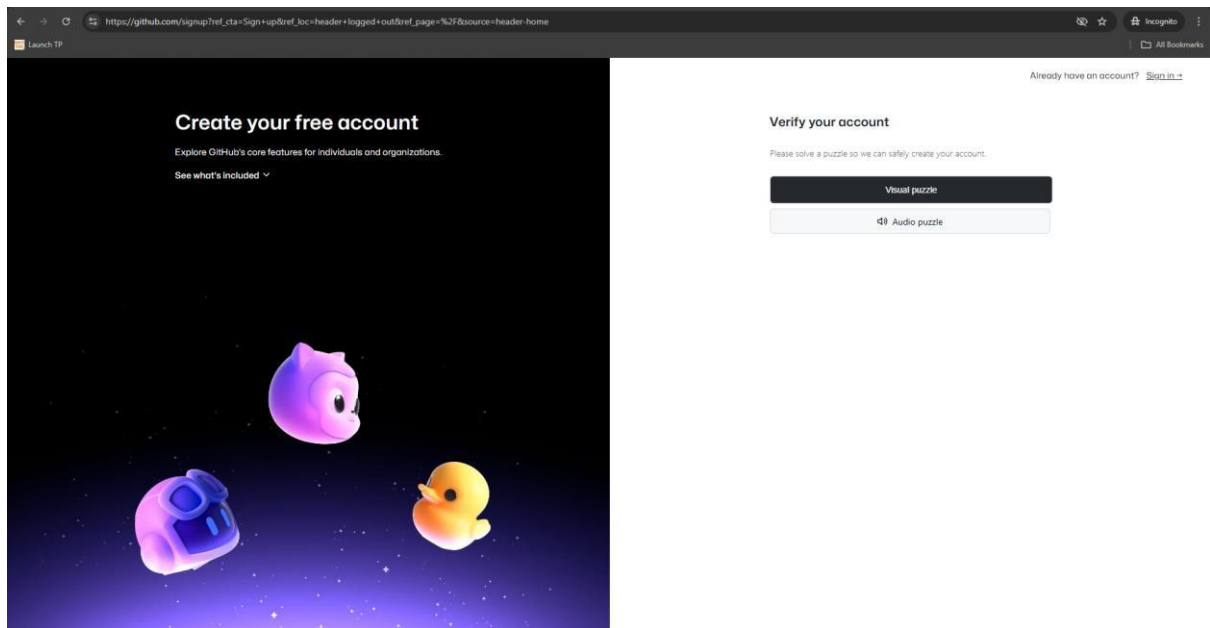
1. Go to <https://github.com/join> in a web browser.

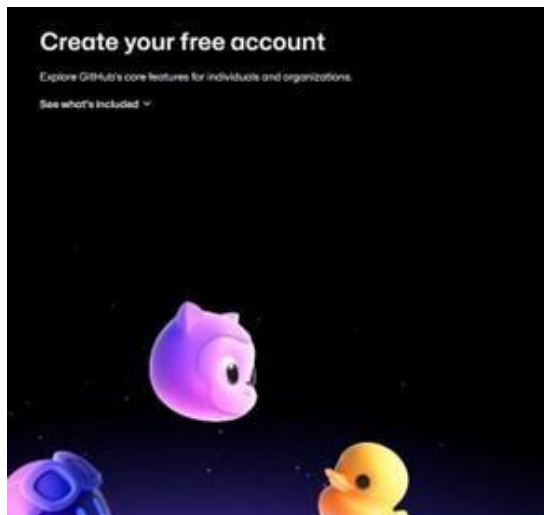


2. Personal Information:

A screenshot of the GitHub sign-up form. The left side has a dark background with the text 'Create your free account' and 'Explore GitHub's core features for individuals and organizations.' The right side is white and contains the sign-up form. The form has fields for 'Email' (vengurlekarheramb10@gmail.com), 'Password' (masked with dots), and 'Username' (Herambve). There are error messages for the password and username. The password message says 'Password needs a number and lowercase letter. Password should be at least 15 characters OR at least 8 characters including a number and a lowercase letter.' The username message says 'Username Herambve is not available. Herambve-igtm, Herambve-tech, or Herambve977 are available. Username may only contain alphanumeric characters or single hyphens, and cannot begin or end with a hyphen.' There is a 'Your Country/Region' dropdown menu set to 'India'. At the bottom, there is an 'Email preferences' section with a checked box for 'Receive occasional product updates and announcements'. A 'Continue' button is at the bottom right.

### 3. Verify email with code:





4. Select your preference plan:

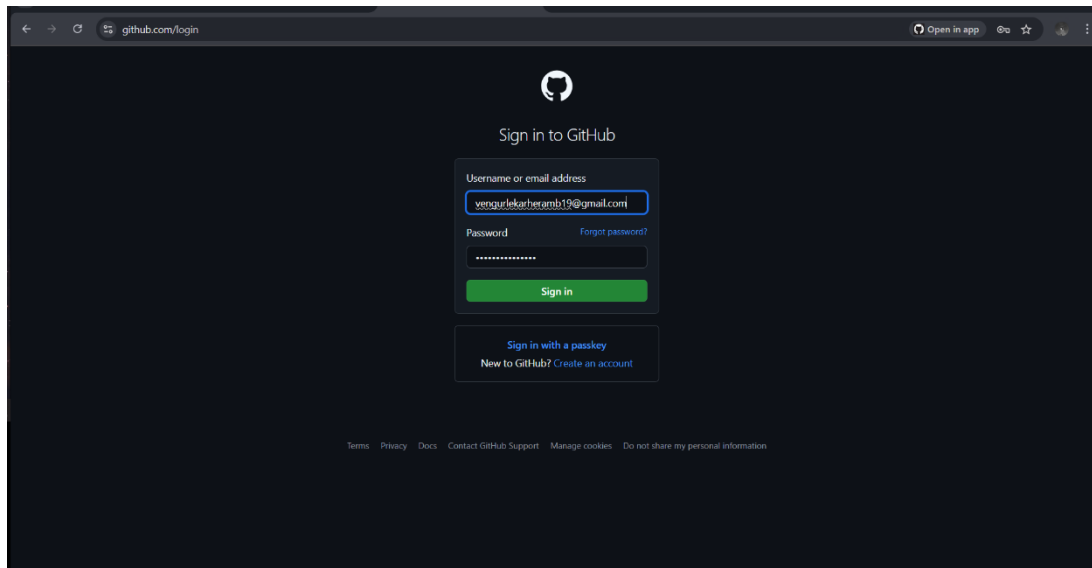
Choose a plan

## Pick a plan for your team

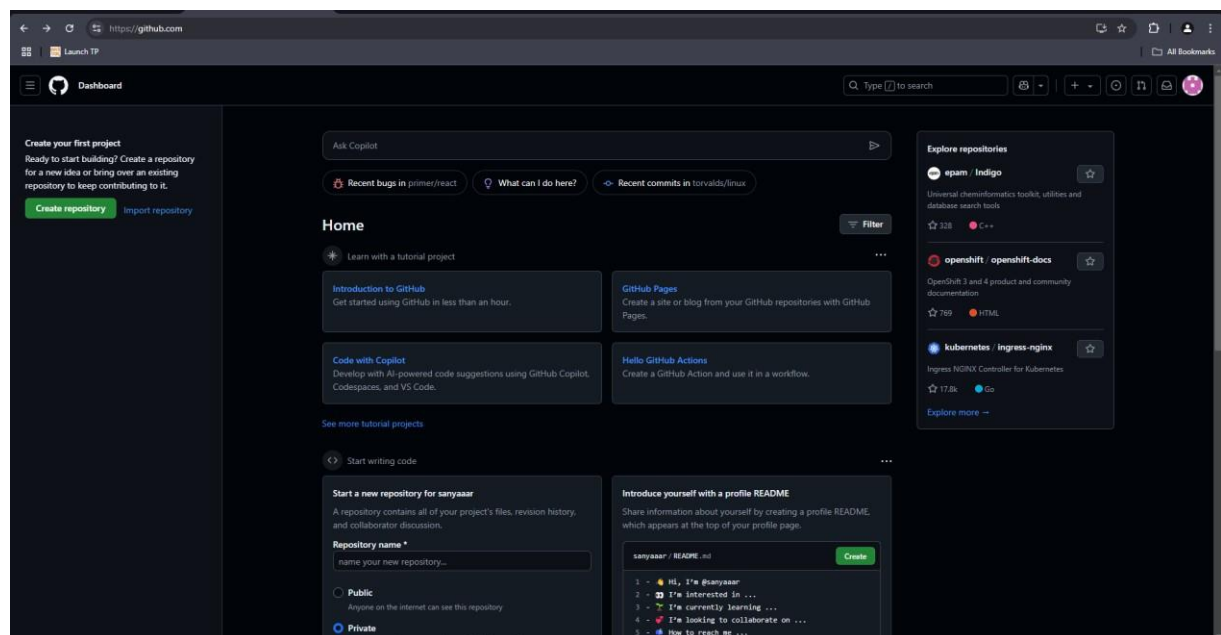
Free	Team	Enterprise
The basics of GitHub for every team	Advanced collaboration and support for teams	Security, compliance, and flexible deployment for enterprises
<ul style="list-style-type: none"><li>Unlimited public/private repositories</li><li>Unlimited collaborators</li><li>2,000 Actions minutes/month Free for public repositories</li><li>500MB of GitHub Packages</li><li>Community Support</li></ul>	<ul style="list-style-type: none"><li>Everything in Free</li><li>Required reviewers</li><li>3,000 Actions minutes/month ⓘ Free for public repositories</li><li>2GB of GitHub Packages</li><li>Code owners</li></ul>	<ul style="list-style-type: none"><li>Everything in Team</li><li>SAML single sign-on</li><li>50,000 Actions minutes/month Free for public repositories</li><li>50GB of GitHub Packages</li><li>Advanced auditing</li></ul>
<b>\$0</b>	<b>\$4</b> per user/month	<b>\$21</b> per user/month
Join for free	Continue with Team	Start Enterprise trial

Heramb Vengurlekar  
T23 – 120

## 5. Sign-in:

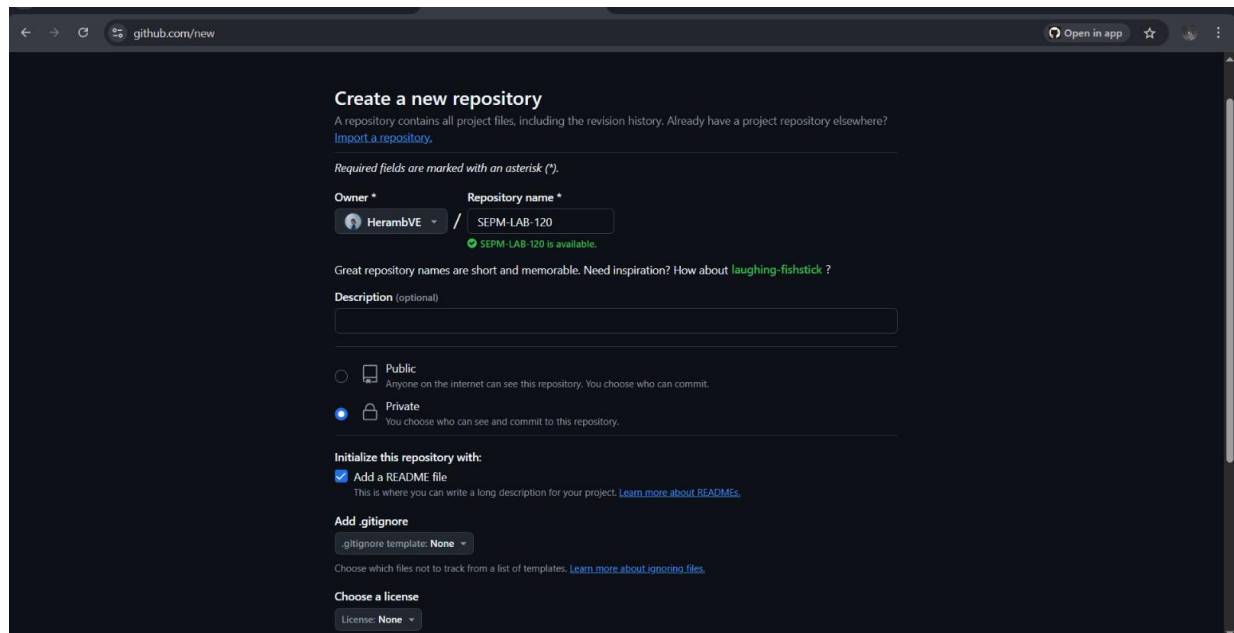


## 6. GitHub Dashboard:



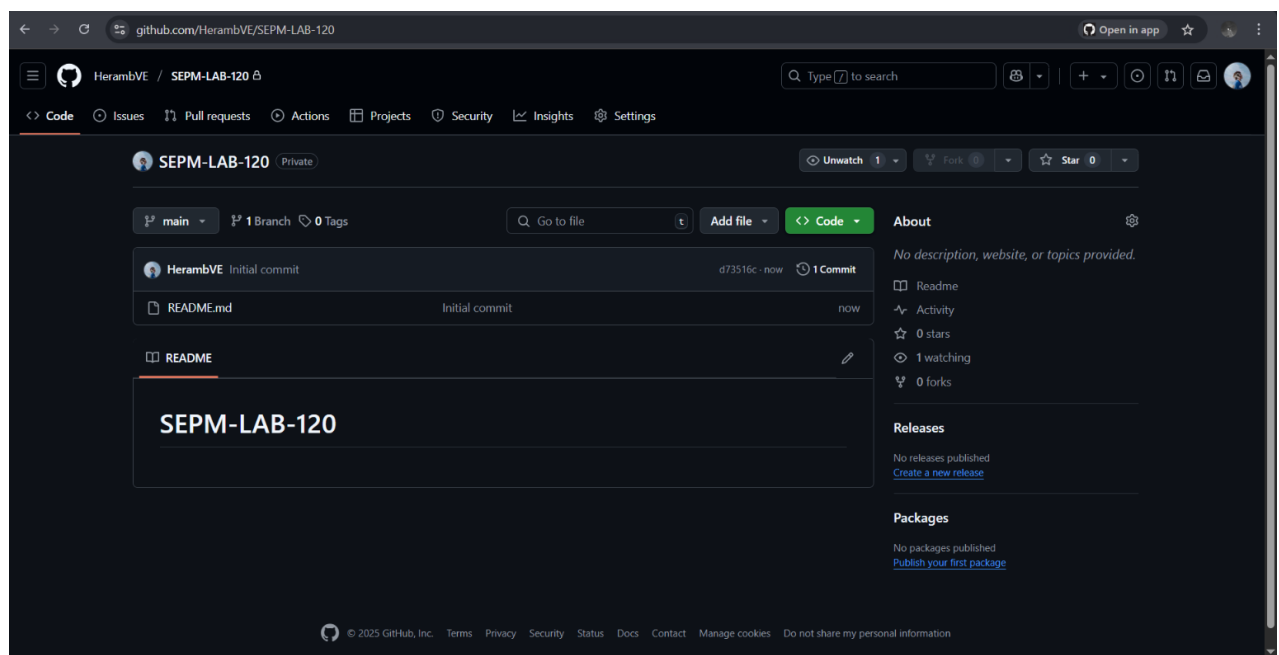
Heramb Vengurlekar  
T23 – 120

## 7. Create New Repository:



The screenshot shows the GitHub 'Create a new repository' page. The page has a dark theme. At the top, it says 'Create a new repository' and 'A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)'. Below this, it says 'Required fields are marked with an asterisk (\*)'. The 'Owner' is set to 'HerambVE' and the 'Repository name' is 'SEPM-LAB-120', with a green checkmark indicating 'SEPM-LAB-120 is available.'. There is a text input field for 'Description (optional)'. Below that, there are two radio buttons for 'Public' (selected) and 'Private'. Under 'Initialize this repository with:', the 'Add a README file' checkbox is checked. There is a dropdown for 'Add .gitignore' with 'None' selected. At the bottom, there is a 'Choose a license' dropdown with 'None' selected.

## 8. Repository Dashboard:



The screenshot shows the GitHub repository dashboard for 'SEPM-LAB-120'. The page has a dark theme. At the top, it says 'HerambVE / SEPM-LAB-120'. Below this, there are tabs for 'Code', 'Issues', 'Pull requests', 'Actions', 'Projects', 'Security', 'Insights', and 'Settings'. The 'Code' tab is selected. The repository is marked as 'Private'. There are buttons for 'Unwatch', 'Fork', and 'Star'. Below this, there is a section for 'main' branch with '1 Branch' and '0 Tags'. There is a search bar 'Go to file' and buttons for 'Add file' and 'Code'. The main content area shows the 'Initial commit' by 'HerambVE' with a commit hash 'd73516c' and a timestamp 'now'. Below this, there is a 'README' section with the text 'SEPM-LAB-120'. On the right side, there is an 'About' section with 'No description, website, or topics provided.' and a list of statistics: 'Readme', 'Activity', '0 stars', '1 watching', and '0 forks'. Below this, there is a 'Releases' section with 'No releases published' and a link 'Create a new release'. At the bottom, there is a 'Packages' section with 'No packages published' and a link 'Publish your first package'.

## CONCLUSION:

Thus, we have successfully created a GitHub account and a new repository in it.