Scenario 1 Report

Group 13: Fangming Luan, Celikten Celikten, William Goodwin, Rida Zaman

Project: *TechLens*, An Intelligent Aggregation Platform for Tech Trends

1. Introduction

In the era of information explosion, technology and software development are evolving at an extraordinary pace. New frameworks, libraries, technologies, and open-source projects emerge almost daily. Some of these further society's technological progress, while a rare few cause such breakthroughs that redefine entire fields and change our daily lives. However, how can individuals proactively discover and keep up with key trends and seize new opportunities in this overwhelming flood of information?

Despite the power of search engines like Google and Bing and the recommendation algorithms of platforms like TikTok and YouTube, the information we receive remains secondhand, passive, fragmented, and dispersed across multiple sources, making it difficult to track and manage efficiently. This often traps individuals within an information cocoon, confining them to what they can easily find and access, limiting their range of topics, preventing them from exploring new horizons, and blinding them to what lies in the ever-changing universe of information and technology.

While open-source communities like GitHub bring us closer to cutting-edge technology, their complex search syntax and relatively high entry barriers make them less beginner-friendly and inaccessible to less technical users. Because navigating and finding valuable projects is quite challenging, many individuals, especially those new to the field, struggle to discover quality content that aligns with their interests. Therefore, we hope to lower the barriers for ordinary people to obtain information and learn about technology and build a **technology-focused information aggregation platform** that reflects some hotspots and trends in the technology field through data analysis and visualization and present them in a dashboard-style interface for a comprehensive overview. Furthermore, this website will be highly customizable, allowing users to set their preferences and filters to find the items and news that meet their needs and collect and manage them. This platform aims to bridge the gap between passive information consumption and active technology discovery by offering a structured and intuitive interface. If feasible, future iterations of this platform may be extended into a browser start page or plug-in, a VSCode extension, or a desktop component for both PC and mobile devices.

2. Features

- **Cross-platform Data Integration**. The platform aggregates data from multiple popular sources and professional within the industry, including *GitHub, Stack Overflow, Reddit, and Hacker News*, providing users with a centralized hub for tech insights.
- **Dynamic Updates**. The platform can continuously gather and display the newest technical projects, articles, and news, ensuring that users are always up-to-date with the latest trends and developments.
- **Interactive Visualization**. The platform utilizes dynamic visual tools like interactive bar charts, pie charts, line graphs, word clouds, and ladder charts to display the popularity, growth, distribution, and

trends of different projects and topics, respectively. The charts also support zooming in and out, as well as capturing sections, and allows users to download them as PNG files.

- **Customizable Experience**. The platform provides predefined filters and allows users to customize them based on preferences like programming languages, project types, and time ranges.
- **Personalized Favorites**. Users can save their favorite projects, articles, and news to their personal collection. Each item can be tagged with labels, notes, and colors, and organized into folders, addressing the problem of scatted resources and making it easier to manage important content.
- **Multi-device Synchronization**. The platform supports account synchronization, allowing users to access their settings, favorites, and collections across devices for a seamless experience.

3. Detailed Design

Our website aims to provide a unified platform for searching and discovering cutting-edge technology projects across multiple sources, starting with *GitHub, Stack Overflow, and Hacker News* as the core focus. By integrating data from various platforms and communities, we seek to give users a comprehensive view of the latest technological trends and innovations. This approach not only addresses the needs of developers but also aims to bring tech enthusiasts, investors, and industry leaders closer to the projects shaping the future. In the future, this approach can be applied to more websites and even different industry sectors. The following sections illustrate the design and functionality of our platform. The highlighted fields are search keywords and terminologies.

Part 1: (Homepage) One-stop, Multi-dimensional Project Filtering and Visualization

We intended to provide the following filtering criteria for users, which reflect the fundamental attributes and impact of a project. These criteria are also critical metrics in our project evaluation and algorithm design:

Keywords	Description
`language`	Filtered by the primary `programming language` used in the project, such as *Python, JavaScript, Java, C++*,
`topic`	Filtered by hotspot `topic`, such as *web development*, *machine learning*, *blockchain*, *AI*.
`stars count`	The number of `stars` reflects the popularity of a project, indicating how many people find it useful and are willing to follow it. This can be filtered by categories such as "more than 10,000 stars" or "between 5,000 and 10,000 stars"
`forks count`	The number of `forks` indicates how many people have copied the project to their own repositories and made modifications.
`contributors count`	The number of `contributors` reflects how many people have taken part in the project's development, indicating the size and activity of the project's community.
`organization`	Filtered by the `organization` that published the project, such as Google, Facebook, Microsoft

Keywords	Description
`issues count`	The number of `issues` reflects how many problems, bugs, or feature requests have been reported by users, indicating the project's maintenance status and user feedback.
`pull requests count`	The number of `pull requests` reflects how many external contributions have been made to the project, indicating the project's openness and community engagement.
`when was created`	The `creation date` indicates when the repository was first published, allowing users to filter repo based on their age.
`when was last updated`	The `last update date` indicates when the repository was last modified, allowing users to filter repo based on their activity level.

Part 2: Dashboard

This section of the website aims to provide a comprehensive and intuitive overview of trending technologies and popular projects through a variety of interactive visual charts. With real-time updates, users can gain a better understanding of industry trends and current hot topics, helping them stay informed about the latest developments in the tech world. The dashboard could includes the following components:

- **Bar Chart** showing the repositories with the most `stars` created in the past week.
- **Bar Chart** displaying the repositories with the most `forks` created in the past month.
- **Line Chart** illustrating the trend in `programming language` popularity over time.
- **Bar Chart** presenting the most popular projects for each `programming language`.
- **Word Cloud** highlighting the current `trending topics`.
- **Line Chart** tracking the fastest-growing `followers` for contributors and organizations over the past week.
- **Pie Chart** representing the distribution of projects across different `programming languages`.
- **Ladder Chart** displaying the `hot topics` and popular `questions` from Stack Overflow, Reddit, and Hacker News.
- **Global Open-source Map** showing the number, quality, and activity level of open-source projects across different `countries`.

Part 3: Personalized Customization

We've taken into account the different intended user base and their specific needs, creating more precise filtering options and control interfaces based on vertical fields segmentation. Users can also personalize their search page by adjusting, adding, or removing filters and controls as needed. The following are the user groups we have considered along with their needs, followed by some corresponding GitHub repository search algorithms:

- **Programming Beginners/Entry-level Learners**: Interested in tutorials for learning new technologies or concepts, and exploring the official documentation and examples of programming languages, frameworks, or third-party libraries:
    - Well-documented projects with detailed `README.md` files, could contain a significant number of `pdf`, `image`, `video`, or website URLs.
    - Tagged with `tutorial` or `guide`, making it easier to find educational resources.

    - Interactive code files such as `Jupyter Notebook` or `Colab` .
    - Have a `wiki` section.
    - List links to `official documentation` repositories for popular languages such as *Java*, *Python*, frameworks like *React*, *Django*, *FastAPI*, and libraries like *TensorFlow*, *PyTorch*, *Scikit-Learn*.
- **Intermediate-level Students**: Looking for some suitable projects to practice, participate in open source projects to gain experience, prepare for job hunting, or further study in a certain field.
    - Marked with `good first issue` or `help wanted`, highlighting beginner-friendly contribution opportunities.
    - Includes a `CONTRIBUTING.md` file.
    - Actively maintained with numerous `issues` and `pull requests`.
    - Built with popular tech stacks like *React*, *Vue*, *Node.js*, *Django*, *Spring Boot*.
    - Related to `interview`, `LeetCode`, `algorithm`.
- **Investors and Product Managers**: Focused on high-potential projects and teams with strong growth and commercial viability:
    - Labeled with `sponsor`, `funding`, or `invest` tags, indicating the project is seeking commercialization or sponsorship.
    - Demonstrating with rapid growth in `stars`, `forks`, and `contributors`
    - Featuring structured development plans with `roadmap`, `milestones`, or `release schedules`.
    - Added open-source licenses like `MIT`, `Apache`, `GPL`, suitable for commercial applications.
- **Software and Gaming Enthusiasts**: Interested in finding fun, useful software, games, plugins, and mods:
    - Includes `download`, `installation guide`, `deploy`, or `setup` instructions for easy access.
    - Tagged with `mod`, `game`, `plugin`, `extension`.
    - Packed with necessary resource files like `.exe`, `.apk`, `.dmg`, `.dll`, `.env`, `.yaml`, `.jpg`, `.mp4` ,`.obj`.
- **Advanced Developers and Tech Enthusiasts**: Want to keep up with the latest technologies and track trending projects:
    - Covering cutting-edge topics like *LLM*, *AI*, *Web3*, *blockchain*, *NFT*, *DeFi*, *Metaverse*.
    - Including `research papers`, `arXiv`, `conference`.
    - Published by major companies like *Google*, *Facebook*, *Microsoft*, *Amazon*.
    - Updated frequently with multiple version numbers, `experimental`, or `beta` tags.

Part 4: Personal Favorites

This is a section dedicated to each user personally, allowing users to paste the URL of items, videos, articles, websites they are interested in here, and the back-end programme will automatically identify the basic content of the link, and automatically generate a corresponding module containing the website icon and title. This page provides a GUI interface, users can create own folders, attach colour labels, notes; support priority management, such as a limited number of top, according to the number of times the user often clicks on the collection of time sorting; also support for export and import functions.

4. Ethical Analysis / Impact Analysis

1. Since the platform aggregates GitHub data and features, it may divert traffic from GitHub, Hacker News, and similar sites.

- **Mitigation**:
  - All interactive elements will include direct links to the corresponding pages, ensuring users visit source pages for detailed project information.
  - The platform will not host any content; it will only provide a summary and visualization of the data.

2. Frequent API call requests and data capture may cause some pressure on these website servers.

- **Mitigation**:
  - The website will strictly adhere to API rate limits and implement caching and request delay mechanisms to reduce redundant queries.
  - User request limitations, such as enforced cooldown intervals between requests, will be introduced to prevent excessive traffic.

3. Calling the GitHub API to get project information may involve users' personal information, and data capture may violate the GitHub usage policy; storing data about personal contact information, social accounts, etc., on GitHub may involve privacy issues.

- **Mitigation**:
  - The website will comply with GitHub's API usage policy, ensuring that data is used only for non-commercial, educational, and research purposes.
  - Do not display or store private information such as personal contact information; only display a project's name, description, and evaluation parameters.

4. Flawed screening logic and algorithms may cause short-term hotspots to be over-amplified, exacerbate technology bubbles and the herd effect, and lead to users lacking judgment to blindly learn technologies that may be rapidly phased out or have limited application.

- **Mitigation**:
  - Get the evaluation and comments of some media and professionals for a project to increase users' understanding of the project and reduce the number of blind followers.
  - Label 'emerging technology' and 'mature technology' to distinguish between short-term hotspots that appear out of nowhere and long-term stable and ecologically mature technologies.

5. Technical Architecture

Data Fetching

To ensure efficient and reliable data retrieval, we prioritize using official APIs whenever possible:

- Utilize `requests`, `httpx` to fetch up-to-date public data from GitHub, Stack Overflow, Reddit, and Hacker News via their respective **APIs**.
- For platforms that do not provide APIs, leverage **Python web scraping** such as `BeautifulSoup` and `Scrapy` to extract relevant content.
- Implement `Celery` for asynchronous task scheduling and background processing, ensuring that data is updated at regular intervals (e.g., every hour or daily) without affecting real-time user interactions.

Data Processing

Once the raw data is collected, it undergoes thorough processing to ensure consistency and usability:

- Use Python's built-in `json` module and `BeautifulSoup` to parse API responses and extract relevant data from `HTML tags` when necessary.
- Employ `Pandas` and `NumPy` for data cleaning, transformation, and aggregation, preparing the dataset for analysis and visualization.

Data Storage

A well-structured storage system ensures fast data retrieval and efficient management of user preferences and historical trends:

- `MySQL` / `PostgreSQL` as the primary relational database for storing user favorites, filtering conditions, and historical data trends.
- `Redis` for caching frequently accessed data, significantly improving response times for popular queries.

Web Frontend

- Languages: **HTML**, **CSS**, **JavaScript**.
- Framework:
  - **React.js** for creating dynamic and interactive UI components.
  - **Bootstrap** for styling design and layout.
  - **Plotly.js** for generating interactive visualizations like line charts, pie charts, and word clouds.

Web Backend

- Languages: **Python**.
- Frameworks:
  - **Django** for user authentication, session management, and seamless frontend-backend communication.
  - **FastAPI** for high-performance data queries and recommendation algorithms, ensuring fast response times for complex filtering operations.

Architecture Diagram