Intersect Browser: Requirements & Team Policies

Team info & policies

a) Mohammad Tanzil Idrisi: AI and Backend Engineering
b) Faiq Ahmad: ROLE
c) Abhas Oli: ROLE
d) Rahul Bikram Shah: ROLE
e) Hamna Zulfiqar: ROLE

GitHub Link: Intersect
Communication Tools: Trello, Whatsapp

Product description

Introducing Intersect—an advanced AI-powered browser that redefines how you search and work. With its integrated AI OS, you can effortlessly manage files and launch apps directly within the browser. Each search triggers a split-window view, delivering rich, in-depth results including videos, images, responses, locations, and accurate citations. Intersect also features AI co-workers to help with research, analysis, and various tasks—all running in the cloud. Optimized for smaller systems, it enhances both interface experience and overall productivity, offering a seamless, smarter approach to browsing and task management.

Features:

1) Search Engine
2) No ads!
3) AI-Powered Bot in Split Window (In-depth search engine)
4) Integrated OS i.e. File and App Management

Stretch goals:

1) Co-working AI agents
2) Integrating all the features of a modern browser

Use Cases

USE CASE 1

Goal	Patron wishes to search for a webpage
Primary actor	Patron
Scope	Web Browser
Level	User
Precondition	Patron is at the welcome screen
Success end	The webpage is loaded and its contents are displayed
Failure end	Failure to load the webpage
Trigger	Patron presses the enter button or the search icon on the sidebar
Main Success Scenario	1. Patron opens the browser 2. Browser presents a welcome screen with a sidebar 3. Patron enters URL or any keyword to search on the search bar 4. Browser finds the webpage and loads the contents
Extensions (error scenarios)	2a. No Internet 2a. 1 Browser loads 'no-internet' page 2a. 2 Patron hits the reload button and tries again 4a. The webpage is not found 4a. 1 Browser loads the 'Error 404 Not Found' page 4a. 2 Patron hits the reload button or backs out
Variations (alternative scenarios)	3. Patron might click the pages already present in the bookmark or the history page

USE CASE 2

Goal	Patron wishes to open a file
Primary actor	Patron
Scope	Web Browser
Level	User

Precondition	Patron is at the welcome screen
Success end	The file is opened.
Failure end	Failure to open the file
Trigger	Patron clicks the file icon on the file explorer
Main Success Scenario	1. Patron opens the browser 2. Browser presents the welcome screen 3. Patron can either enter a file path or click on the file icons 4. Browser loads the file
Extensions (error scenarios)	4a. File Path Not Found 4a. 1 Browser loads the 'Error 404 Not Found' page 4a. 2 Patron hits the reload button or backs out
Variations (alternative scenarios)	3. Patron might click the recent file history present in the browser

USE CASE 3

Goal	Patron wishes to perform an in-depth search
Primary actor	Patron
Scope	Web Browser
Level	User
Precondition	Patron is at copilot sidescreen
Success end	The search results are displayed with proper citations
Failure end	The search results aren't displayed or inaccurate results show up
Trigger	User presses the search icon on the copilot sidescreen
Main Success Scenario	1. Patron opens the browser

	2. Browser presents the welcome screen 3. Patron opens the copilot sidescreen 4. Patron enters keyword to search on the search bar 5. The AI search engine loads the results with proper citations
Extensions (error scenarios)	2a. No Internet 2a. 1 Browser loads 'no-internet' page 2a. 2 Patron hits the reload button and tries again 5a. Inaccurate Results 5a. 1. AI search engine provides inaccurate results with wrong citations. 5a. 2. Patron performs the search again or backs out.
Variations (alternative scenarios)	

USE CASE 4

Goal	Patron wants to save a webpage
Primary actor	Patron
Scope	Web Browser
Level	User
Precondition	Patron is at a webpage
Success end	The webpage is bookmarked
Failure end	The webpage is not bookmarked
Trigger	Patron clicks the bookmark icon at the menu bar
Main Success Scenario	1. Patron opens the browser. 2. The browser displays the welcome screen. 3. Patron searches for a webpage. 4. The browser loads the webpage. 5. Patron clicks the bookmark icon. 6. The browser bookmarks the page.

Extensions (error scenarios)	2a. No Internet 2a. 1 Browser loads ‘no-internet’ page 2a. 2 Patron hits the reload button and tries again 4a. The webpage is not found 4a. 1 Browser loads the ‘Error 404 Not Found’ page 4a. 2 Patron hits the reload button or backs out 6a. The webpage is not bookmarked. 6a. 1 Patron tries again or backs out.
Variations (alternative scenarios)	5. Patron clicks the keyboard shortcut (Cmd + B)

USE CASE 5

Goal	Patron wishes to open the terminal
Primary actor	Patron
Scope	Web Browser
Level	User
Precondition	Patron is at the welcome screen
Success end	The terminal is opened
Failure end	Failure to open the terminal
Trigger	Patron searches ‘Terminal’ through the searchbar in the sidebar
Main Success Scenario	1. Patron opens the browser. 2. Browser displays the welcome screen. 3. Patron searches ‘Terminal’ on the search bar in the sidebar. 4. The browser opens the terminal.
Extensions (error scenarios)	2a. No Internet 2a. 1 Browser loads ‘no-internet’ page 2a. 2 Patron hits the reload button and tries again 4a. Failure to open terminal

	4a. 1 Patron tries to search again or backs out.
Variations (alternative scenarios)	3. Patron clicks the terminal app in the recent apps section.

Non-functional Requirements

1) Loads webpages quickly, minimizing response times for user actions.
2) File explorer system protects user data.
3) The interface should be intuitive and user-friendly.

External Requirements

1) The browser should be compatible with all operating systems.
2) The browser must support third-party extension or plugins.
3) The browser must adhere to ISP-imposed restrictions.

Team process description

Software Toolset: Electron.js, React.js, HTML, CSS, JavaScript,

Roles:

1) Mohammad Tanzil Idrisi, Backend Developer & AI Engineer: We need to develop an in-depth AI search engine with citations and AI coworkers. As he has previous experience in designing AI models and full-stack development, he will be working on these areas of our project.
2) Faiq Ahmad, FullStack Developer: We need to integrate OS functionality such as file and app management in the browser. As he has previous experience in OS architecture and backend development, he will be working on these areas of our project.
3) Abhas Oli, FullStack Developer: We need to develop the functionalities of browsers and seamlessly integrate the back end with the front end. As he has previous experience in full-stack development, he will be working on these areas of our project.
4) Rahul Bikram Shah, FrontEnd Developer: We need to design and build the front-end components of our browser. As he has experience with UI/UX design and frontend development, he will be working on these areas of our project.
5) Hamna Zulfiqar, FrontEnd Developer: We need to design a seamless and responsive user interface for our browser. As she has experience with UI/UX design, she will be working on these areas of our project.

As we move forward with fulfilling our milestones, our roles might look a bit different.

Project Schedule/Milestones

	Tanzil	Faiq	Abhas	Rahul & Hamna
1	Integrating LLM	File Explorer Path System	Integrate search engine functionality in our sidebar search bar	Design & build a component for the welcome page.
2	Integrating Brave Search	Recent File/Folders	User Profile Login & Authentication	Design & develop buttons, input fields, textbox, & forms.
3	Integrating the Serper API for followup question	Drag/Drop File Explorer	Browser Menu Items (File, Edit, View, History)	Design page frame for browser feature
4	Integration Stable diffusion model API for generating image as well	Integrate search functionality	Adding keyboard shortcuts and extensions	Design page frame for AI features.

Three major risks:

1) Finding an efficient way to integrate OS functionality in our browser.
2) Incorporating all the features of a modern browser due to the time constraints.
3) Integrating AI coworker feature in our browser.

Receiving Feedback:

We will be constantly looking for feedback from our classmates, professors, and general users after completing each milestone. We will let them use our features in the browser at that moment and receive constructive criticism to improve those in the next iteration.