Software Engineering

CSC648/848

Milestone 1

Section 04 | Team 2

Artemis

2/28/2022

Brian Adams	badams2@mail.sfsu.edu	Team Lead
Joseph Kois	jkois@mail.sfsu.edu	Github Master
Brandon Cruz-Youll	bcruz5@mail.sfsu.edu	Scrum Master
Sherman Wong	swong42@mail.sfsu.edu	Backend Lead
Arslan Alimov	aalimov@mail.sfsu.edu	Backend Support
Jade Simien	jsimien@sfsu.edu	Frontend Lead

History Table:

2/14/2022 - Everyone together as a team to create an executive summary, functional and non-functional requirements, personas, and competitive analysis.

1.Executive Summary:

Artemis is a bug tracking application that will empower developers to track new and existing bugs. Being able to efficiently create and delegate responsibilities from a single dashboard enables development teams to prioritize and resolve issues, coordinate with quality assurance teams, and plan for sprints. Artemis condenses what the development team needs to know about issues and technical debt, in an easy-to-use application.

Artemis will provide a roles-based system that gives various users different privileges. A quality assurance team would be able to create tickets, a scrum master would be able to delegate issues, and developers would be able to mark tickets as being closed. This offers various teams the ability to work together, while reducing the possibility for error. Users will be able to see a changelog of the project history to understand what exactly has been changed with the program. Also, users will be able to view a roadmap of the current project they are assigned to, that way they can get an idea of what's to come.

Artemis strives to be a minimally designed, efficient, and effective project dashboard for tracking bugs. The ease of use throughout the system propels it above its competitors. The problem Artemis tries to solve is the issue of feature creep and bloat. Many other bug trackers will have features that are not necessary and oftentimes get in the way. Having a cluttered UI can make for a frustrating experience. Artemis strives to solve this with a minimalist design and easy-to-use features.

2.Personas:

CEO of startup

Age: 40

About:

- She has been in the field for a while and has decided to take the plunge and start her own company.
- She has a Masters in Computer Science and did research in Machine Learning and Natural Language processing.
- She has worked for well-known companies such as Google and Apple, further developing her skills in the field.
- She is known for her contributions to the field–including her research.

Attitude:

- She is a go-getter but still has empathy for others around her.
- She helps lift people up, rather than see them as stepping stones to better places.

- She has always been somewhat of a risk-taker, but embarking on this startup adventure has led her to be more adventurous.
- She roots for the underdog and seeks to help other startups that she sees as great innovators.

Skills:

- Fluent in Python, R, JSON, Java, C, C++
- Expert in voice recognition and language modeling

Goals:

- Continue developing start up and work on getting VC funding

Software Developer

Age:27

About:

- He came from another country to pursue his dream career in the United States.
- He is knowledgeable and has completed his education, such as Bachelors in Computer Science
- He is on his third job, however, he gained a lot of experience from his current job. He learned a lot about backend and front end development. He gained a lot of experience, product design knowledge and development skills from his past jobs.
- He loves challenges, teamwork, communication, and complicated tasks.

Attitude:

- Hard-Working, loves to learn new things especially in coding.
- Patient with people, especially his co-workers.
- Loves to listen and criticize work of his own and his teammates.
- Always positive and fun to interact with.

Skills:

- Fast Typing
- Knowledge of languages such as: Java, Python, Javascript, JSON, C,C++, C#, Swift

Goals:

- Wants to become rich by 30 years old.

College Intern (Software Developer)

Age: 22

About:

- Hard working student their whole life, very excited to get into the world of software development.
- Their first time working at a startup.
- They are in their last semester of college.

- Has never used software development tools before.
- They are interested in creating their own startup company someday.

Attitude:

- Frequently pulls all-nighters to debug code.
- Very sluggish in the morning, works best at night.
- Can be irritable sometimes when they forget to take breaks and eat.

Skills:

- Fluent in C++, CSharp, Java, Typescript
- Computer Networking
- Socializing

Goals:

- Create their own startup someday.
- Be the next Jeff Bezos.

3. Data Definitions:

Bug: Issue with software
Ticket: Single Bug, or point of technical debt
Close: To resolve a ticket
Workspace: A space where all the bugs for a specific portion of a project will be collected
Team: Collection of individuals with specific permissions in any given workspace
Organization: Collection of teams that belong to a workspace

4. Initial list of functional requirements:

- view tickets which act as issues or TO-DOs
- create tickets
- view a roadmap of the project
- see a changelog of previous versions and changes
- see previously resolved issues

5. List of non-functional requirements:

- Require users create an account with secure password
- Minimal design to promote efficiency and usability
- Support HTTPS
- Maintainability with all components of the program

6. Competitive analysis: Jira + Trello

Jira +Trello	Artemis
Roadmap feature to organize the duration of development	Simple interface that is designed with productivity in mind
Agile reports that track time management, created issues, etc.	Tracks issues, and the state of the issues through development
Integrates a team’s commonly used apps into the workflow	Stand-alone application that does not require any third-party integration
Checklists for sub-tasks	Issues can be prioritized and be in various states
Can leave comments for the team or private notes on cards	Devs can comment on the bugs, and list blockers

Jira and Trello approach project management with superfluous designs and features that can be overwhelming. Artemis is planned to have a simple interface and result in a more intuitive workflow. While Trello allows checklists for sub-tasks, Artemis organizes tasks/issues based on priority levels. This encourages productivity by allowing teams to focus on the most pressing tasks to be completed rather than overwhelming the flow with a plethora of low leveled tasks to be checked off.

7.High-level system requirements:

** No. refers to the item number & priority of item*

No.	**Requirement**	**Description**
01	View Ticket	Able to view tickets that were created
02	Create Ticket	Create tickets that need to be attended to
03	Roadmap	View a roadmap of the current project or future projects
04	Changelog	Keep track of all changes made to the codebase
05	Resolved issues	Be able to see previously completed tickets
06	Username & password	Give users the ability to sign up/ log-in, in a secure manner

07	HTTPS	Be able to support HTTPS for secure browsing/use
08	Design	Minimal design to promote efficiency and usability
09	Maintainability	All components structured in a way so that maintainability is easy

8. High Level frameworks and tools

Backend:

- Express
- TypeScript/JS
- TypeORM/Sequelize
- MySQL
- Session
- Bcrypt
- UUID

Frontend:

- TypeScript
- NextJS or React
- CSS Modules
- SWR
- Zustand
- Framer Motion
- UUID?

Design:

- Figma

Version Control:

- Git w/ GitHub

Testing:

- Postman e2e
- Selenium e2e
- Jest unit test

9. Team:

Brian Adams	badams2@mail.sfsu.edu	Team Lead
Joseph Kois	jkois@mail.sfsu.edu	Github Master

Brandon Cruz-Youll	bcruz5@mail.sfsu.edu	Scrum Master
Sherman Wong	swong42@mail.sfsu.edu	Backend Lead
Arslan Alimov	aalimov@mail.sfsu.edu	Backend Support
Jade Simien	jsimien@sfsu.edu	Frontend Lead

10 .Checklist:

- Team found a time slot to meet outside of the class Done
- Scrum Master shares meeting minutes with everyone after each meeting. Done
- Github master chosen Done
- Everyone sets up their local development environment from the team's git repo. Done
- Team decided and agreed together on using the listed SW tools and deployment server Done
- Team ready and able to use the chosen back/front-end frameworks. Done
    - For each technology (front/back-end/DB/cloud), the team decides who will lead the study of each technology and what will be output of the (feasibility) study within one month. On Track
- Team lead ensured that all team members read the final M1 and agree/understand it before submission Done