Sprints (DevOps Tool for students)

Requirements and Specification Document

2023-05-04, v4.0

Project Abstract

Sprints is a planning and tracking DevOps software platform that is built for students and professionals willing to leverage agile methodologies for their development. It provides teams with access to a comprehensive suite of Agile resources to assist in the development of their software projects such as: Sprint planning, Poker, Chatroom, and Educational Resources.

Designed with simplicity and ease of use in mind, Sprints offers an intuitive and user-friendly interface that streamlines collaboration and enables developers to be responsive to changing requirements, reduce project risk, and increase customer satisfaction. With a powerful suite of features and tools, Sprints is the ultimate platform for anyone seeking to build better software faster and more efficiently with Agile methodologies.

Document Revision History

Rev. 1.0 <2023-03-07>: initial version

Rev. 2.0 <2023-04-06>: version 2; modified user requirements, use cases, and user interface requirements

Rev. 3.0 <2023-04-20>: version 3; modified user requirements-added story poker and chatroom and updated task board. Added ERD and database diagrams in the specification. Added screenshots to user interface requirements.

Rev. 4.0 <2023-04-05>: version 4; modified user requirements with updated/corrected features. modified user interface. Increased the functionality and security of the task board. Networking works correctly for story poker.

Customer

Our primary consumers are students who need to build software but do not know Agile or lack experience with software development. This includes all future teams in project focused classes, teams in hackathons, and small businesses. Sprints can also be used by professional teams looking to learn Agile or used Agile methodologies for their projects.

It is a well-known fact that Agile methodology is far superior to several others for developing working software with ability to adapt and overcome hurdles associated with development. However, those looking to develop with Agile are limited to online knowledgebases like YouTube and books. There are very few tools which integrate seamlessly into a developer platform to learn and practice agile on the fly. *Sprints* fills this gap in a multi-billion-dollar industry. It guides teams through Agile in an intuitive way to ensure every team is keeping up with the best Agile practices.

Upon discussing with other students in the class (CS506: Software Engineering), we found out that students find the textbook clunky, annoying and difficult to absorb without practicing Agile methods first-hand. Hence, we have external project interest. We also asked some working professionals (recent graduates) who surprisingly didn't have a lot of knowledge with agile although their team was 'agile'.

User Requirements

- User Stories/Story Poker
  - Users can create a user story using a form.
  - Users can add multiple stories.
  - Users can present a user story as a question to their team.
  - Users can answer questions presented by their team.
  - Multiple users should be able to 'play' story poker at the same time.
  - Participants will answer, in real time, how long a particular story is estimated to take.
  - All participants' results will be shown after answers have been submitted and the team can discuss the results.

- Chatroom
  - Allows for quick and easy communication while planning.
  - All team members can communicate on the same chat.
  - The chat can be used alongside story poker and/or while creating tasks so that if discussion is needed, there is no need to switch platforms.
  - The chat will be sent to the entire team anonymously to encourage team transparency and collaboration.

- Task Board
  - Each task has a “priority rating”, “hours to complete”, and “due date” as a measure of importance.
  - Tasks can be assigned to single/multiple team members.
  - Filtering functionality to enable user to browser through different tasks(priority, deadline, etc)
  - Tasks can be filled out using a form.
  - The form can be made visible/invisible for easy viewing and convenience.
  - Tasks can be modified or deleted.

- Educational Resources
  - Explanation on the home page that explains the overview of *Sprints* and its correspondence to agile methodologies.
  - Users can access the education page to read more about Agile or DevOps.
  - Users can access links on the page as a quick and reliable way to gather more information.

- Having multiple different users
  - Users can log in with google auth.
  - Users’ profile will show up on the navbar.
  - Multiple users can connect with each other in story poker and chatroom features.

Use Cases

- Tasks – **must have (All team)**

  - I want seamless task creation and assignment. My team and I should be able to intuitively navigate the task board. They should be encouraged to ask for clarification if they need specific information (comments/chat).

  - Team members can create their own tasks in the team board. Those assigned to the task must fill in all required fields such as due date, assigned users, title, and priority.

  - I want to be able to delete, edit, and manage tasks.

- Educational resources – **must have** (students, teams, PO, SM)
  - My goal is to easily navigate the tool and learn agile methodologies in a seamless way that doesn’t require hiring an Agile Coach.
    - The tool includes in-house educational resources, tutorials, and guidance for team members who are new to Agile methodologies.

- Story Poker – **must have** (developers, product owner, scrum master)
    - My goal is to provide a simple tool so that Agile teams can conduct story pokers on their user stories in a simple manner. The tool includes a page that allows one person to add a story andevery other team member to add their time estimates.
    - There is an internal timer (1 minute).
    - After everyone is done, the page shows everyone’s estimates in a table.

Name	Task board for project progress control
Actor	User of the Agile app
Trigger	Upon open the navbar and signed in
Events	After signing up with google, user can create a task with desired priority, deadline, and participants
Exit Condition	User signed in and create a task using the UI page
Post-condition	Task successfully created and will be shown in the task list
Acceptance test	User can check all the created task on the bottom of the task board UI page

Name	Story Poker for estimating hours needed for a certain task
Actor	A presenter for the questions, and team’s members for answering
Trigger	When the team lead presents the question using the UI page
Events	- Team lead presenting the question - Team members answered the estimated time needed - Result will be shown with everyone’s answers
Exit Condition	The question is presented, and all the members submit the estimation of hour needed
Post Condition	All the answers from the team will be collected and shown on the result page

Acceptance test	The users can see the result including their own answers on the result page

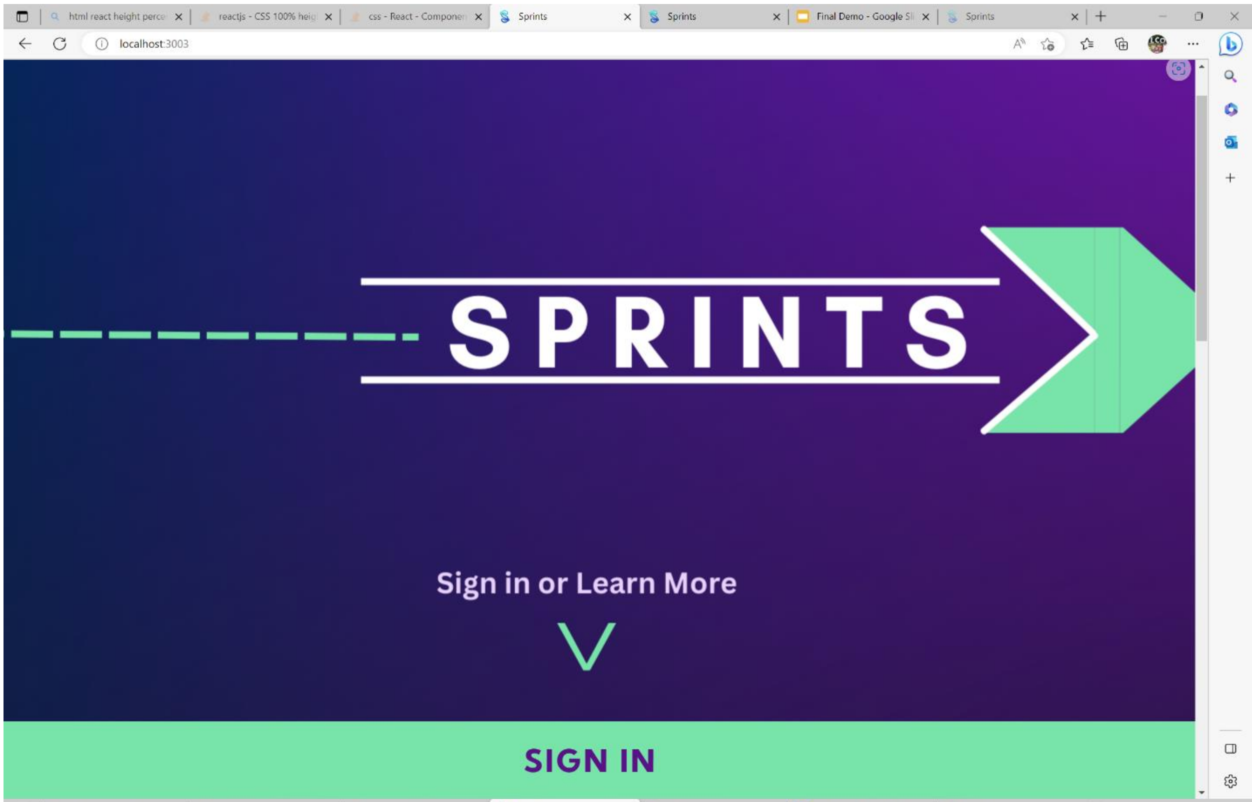
Name	Login page using google sign up
Actor	All users should login to use Agile appfeatures
Trigger	When users click the sign-up UI page
Event	-After user click the sign-up page link, theywill be prompted to sign up using their google account -After selecting the google account, the name and a profile photo will be shown asthe user’s identity
Exit Condition	The users will finish signing up after selected the google account they wish to use
Post Condition	The user will be keep signed up after sign in completed with google auth
Acceptance Test	The user can see their account after signedin in the login UI page

Name	Chatroom for team members to chat online
Actor	All users in the project
Trigger	Upon clicking the navbar for the chatroom features
Event	Users can type in the chat box and see others’ chat at the same time
Exit Condition	After entering the chat room, all the chat will be lived and see by the whole team
Post Condition	Chat can be seemed by the whole team, and after exiting the chat room, the message will no longer shown
Acceptance Test	The live message can be seemed by all the team members immediately after they sent it, as a live chat

User Interface Requirements

- Graphical User Interface Requirements:
  - The interface should have a modern and clean design with consistent layoutand color scheme.
  - The interface will utilize the bootstrap library combined with a CSS doc foreach page.
  - All pages, aside from the home screen should only be revealed when theuser has signed in using google SSO.
  - The task table should include a form that is filled out to add a task and thatform should be able to appear and disappear with a click of the button.
  - The user should be able to share the tables and story poker with other users.
  - When the user updates the tables, it should be updated in real time for allusers.
- Data Exchange Format Requirements:
  - The tool also supports pdf uploads, figma links, git links, and other customizable web links.

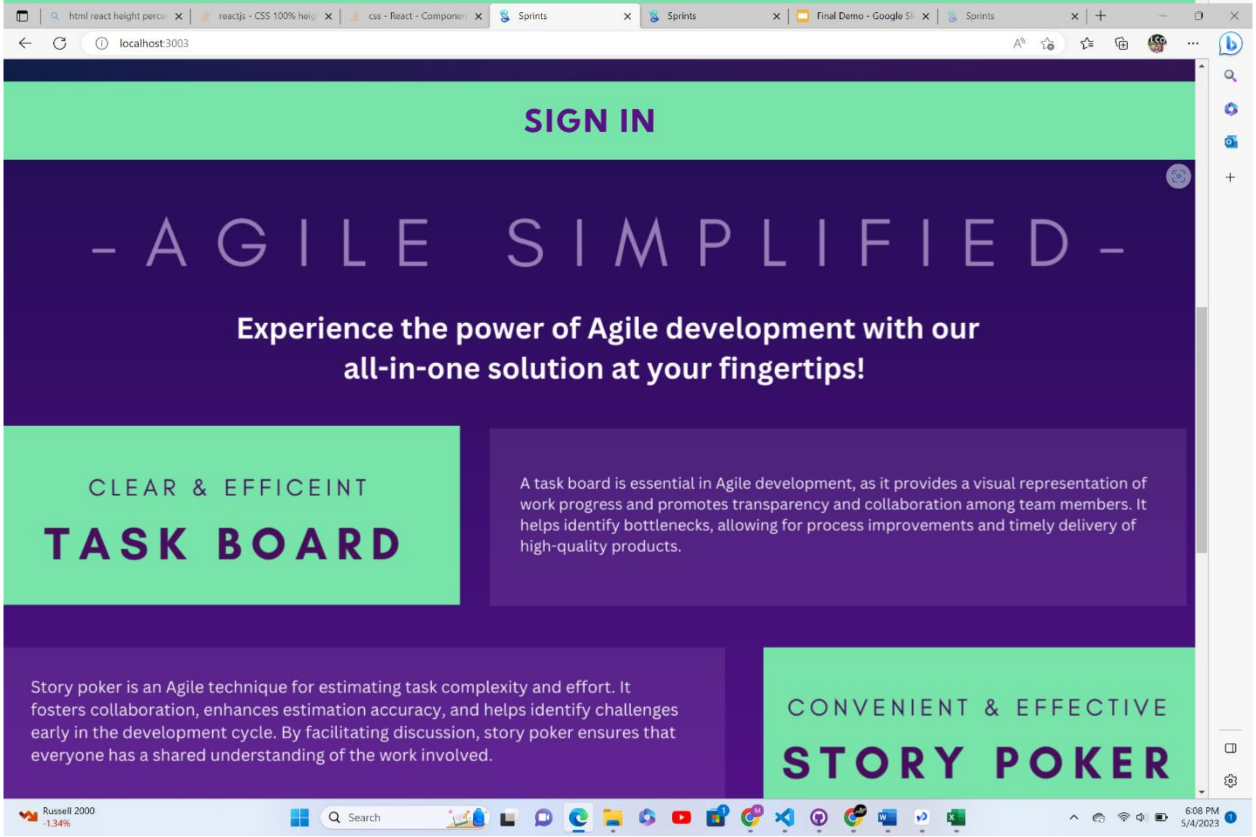
- Reporting and Output Requirements:
  - The platform will report the outputs of the tables and the story poker to allusers.
  - The platform should provide real-time feedback to users on their progress towards completing their projects.
- Workflow Requirements:
  - The platform should allow users to create and manage Sprints, Kanban boards, and backlogs using a form.
  - The platform should provide a collaborative workspace where team members can collaborate on the creation of tasks from the task board and through story poker.
  - The users should be able to add, edit, and delete tasks.
  - The user should be able to sort the different columns in the task board.
  - The platform should provide clear guidance and instruction on how to use each feature and how to apply Agile methodologies to their projects.
  - The platform should have features such as story point estimation and task assignment to enable users to plan and manage their work effectively.



SPRINTS

Sign in or Learn More

SIGN IN



SIGN IN

- AGILE SIMPLIFIED -

Experience the power of Agile development with our all-in-one solution at your fingertips!

CLEAR & EFFICEINT

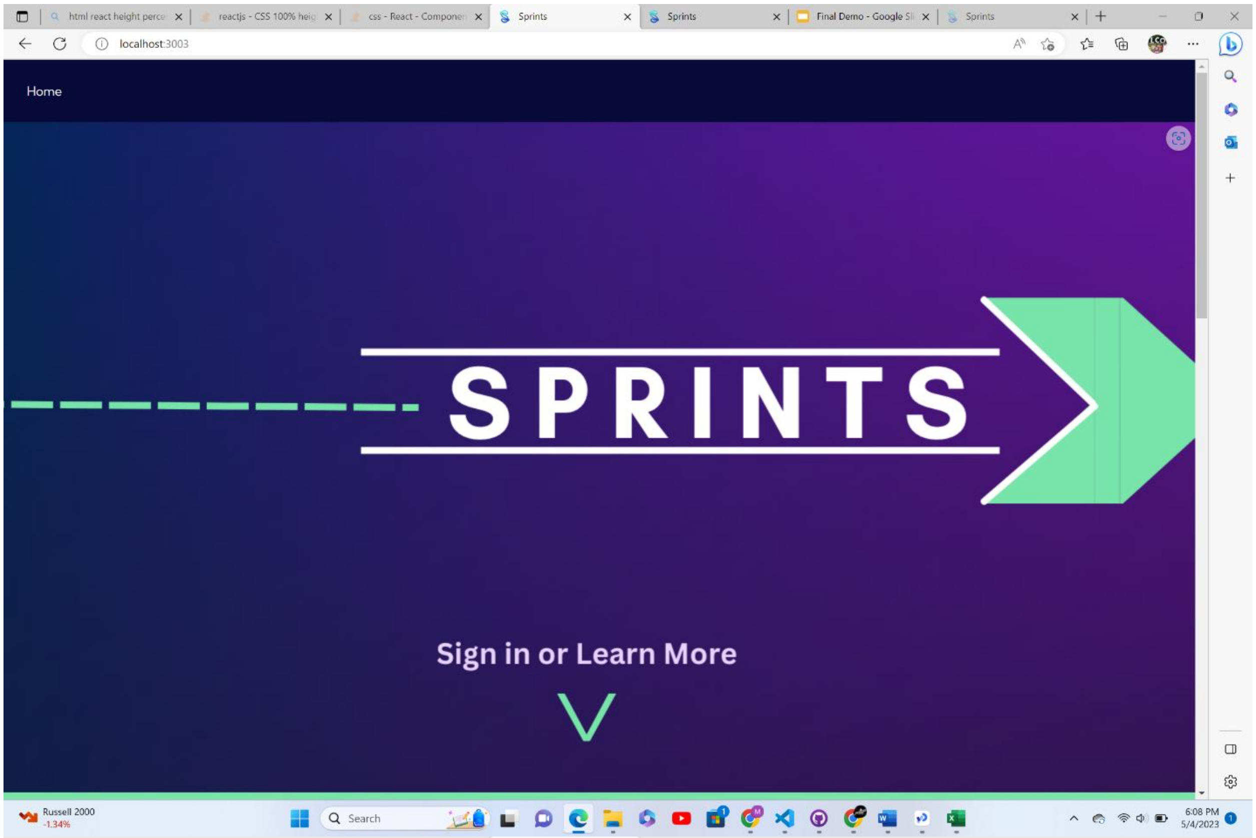
TASK BOARD

A task board is essential in Agile development, as it provides a visual representation of work progress and promotes transparency and collaboration among team members. It helps identify bottlenecks, allowing for process improvements and timely delivery of high-quality products.

Story poker is an Agile technique for estimating task complexity and effort. It fosters collaboration, enhances estimation accuracy, and helps identify challenges early in the development cycle. By facilitating discussion, story poker ensures that everyone has a shared understanding of the work involved.

CONVENIENT & EFFECTIVE

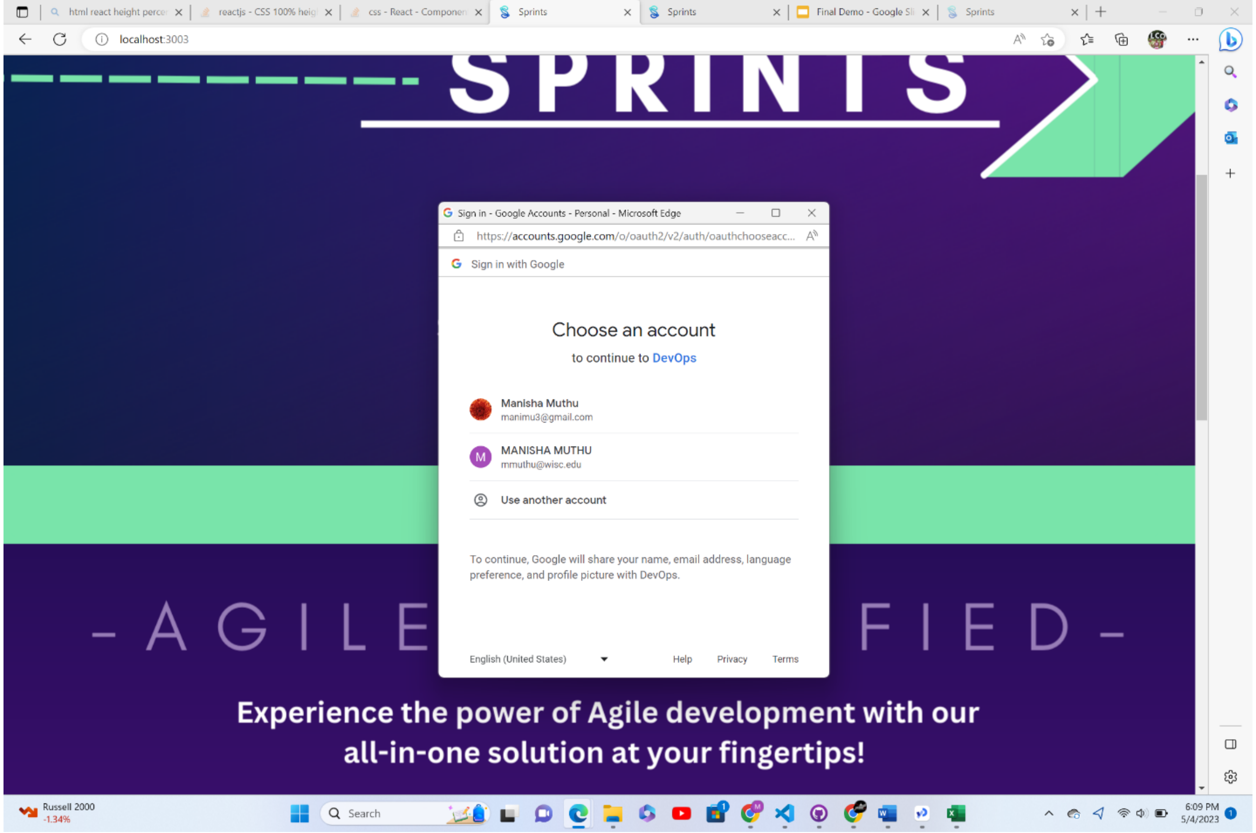
STORY POKER



Home

SPRINTS

Sign in or Learn More



SPRINTS

Sign in - Google Accounts - Personal - Microsoft Edge

https://accounts.google.com/o/oauth2/v2/auth/oauthchooseacc...

Sign in with Google

Choose an account

to continue to DevOps

Manisha Muthu
manimu3@gmail.com

MANISHA MUTHU
mmuthu@wisc.edu

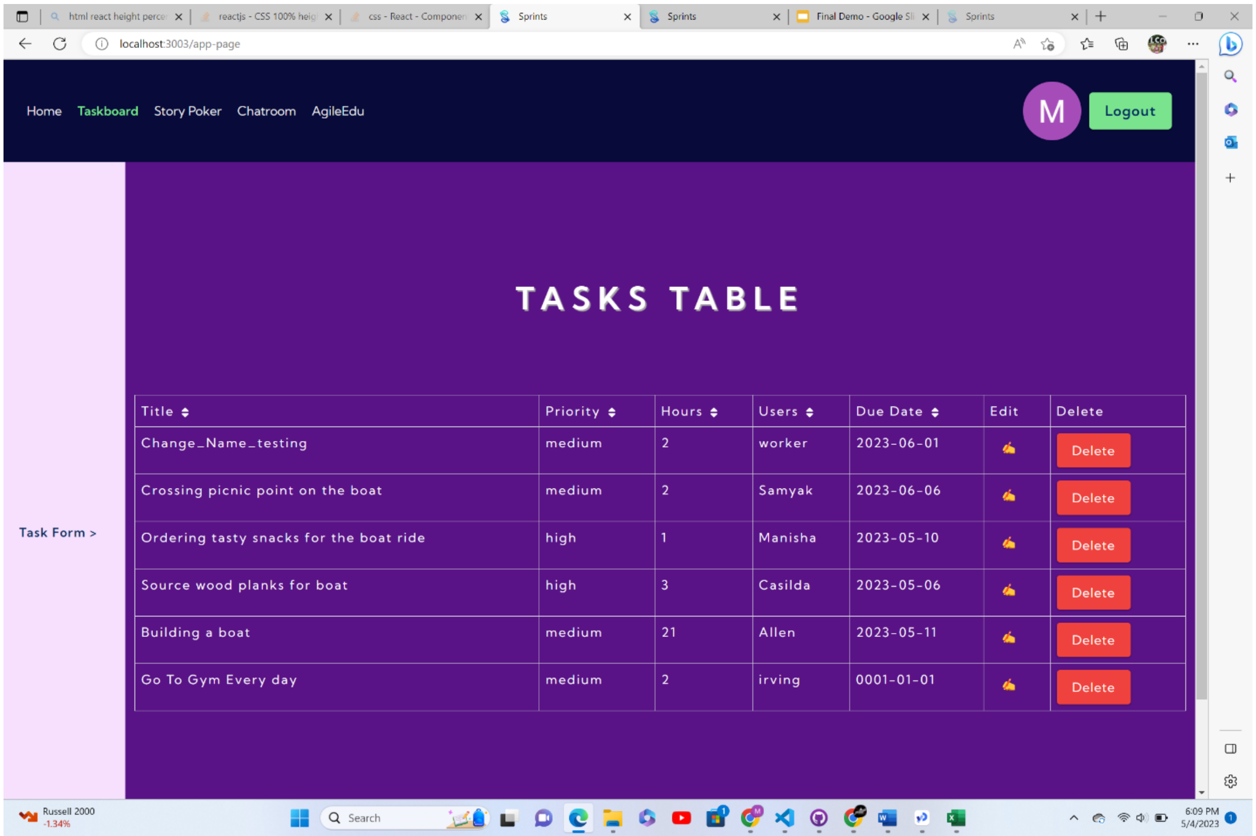
Use another account

To continue, Google will share your name, email address, language preference, and profile picture with DevOps.

English (United States) Help Privacy Terms

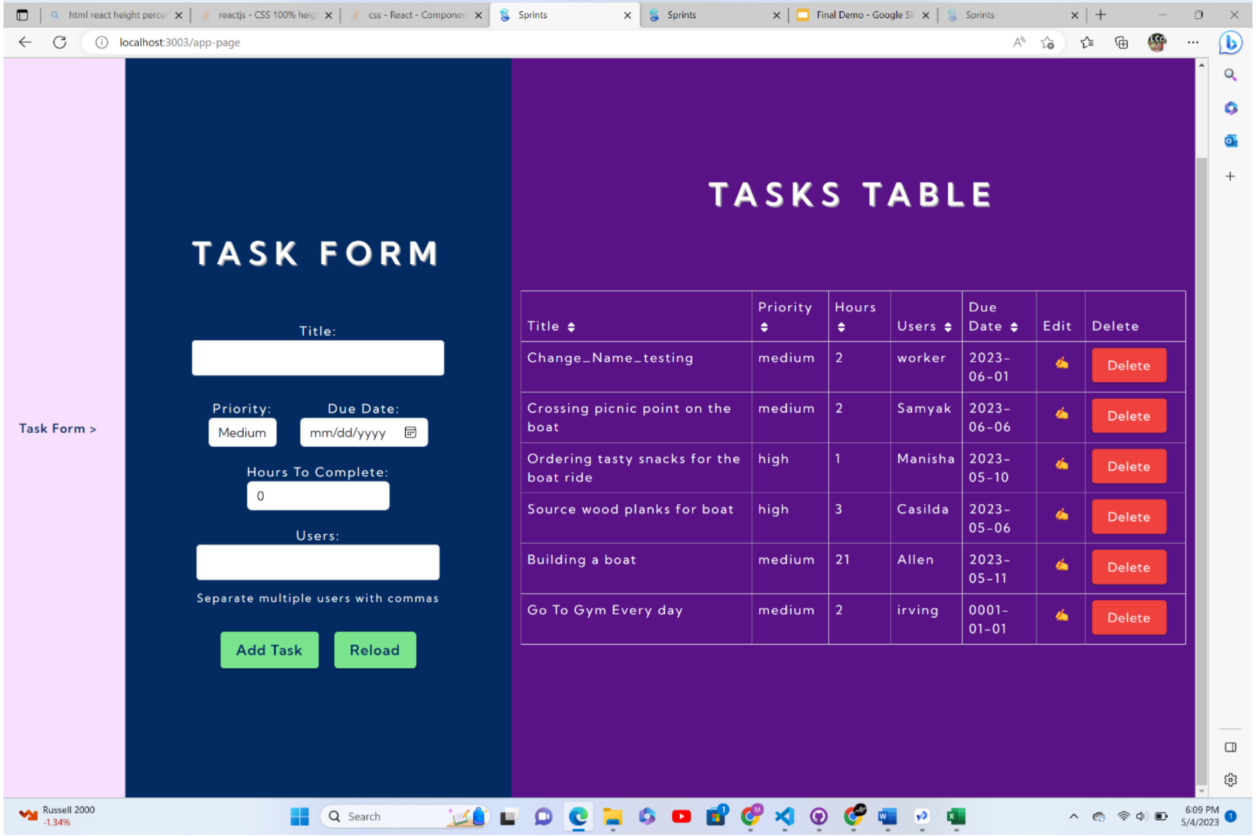
- A G I L E F I E D -

Experience the power of Agile development with our all-in-one solution at your fingertips!



TASKS TABLE

Title ⬍	Priority ⬍	Hours ⬍	Users ⬍	Due Date ⬍	Edit	Delete
Change_Name_testing	medium	2	worker	2023-06-01	✍	Delete
Crossing picnic point on the boat	medium	2	Samyak	2023-06-06	✍	Delete
Ordering tasty snacks for the boat ride	high	1	Manisha	2023-05-10	✍	Delete
Source wood planks for boat	high	3	Casilda	2023-05-06	✍	Delete
Building a boat	medium	21	Allen	2023-05-11	✍	Delete
Go To Gym Every day	medium	2	irving	0001-01-01	✍	Delete



TASK FORM

Title:

Priority: Medium

Due Date: mm/dd/yyyy

Hours To Complete: 0

Users:

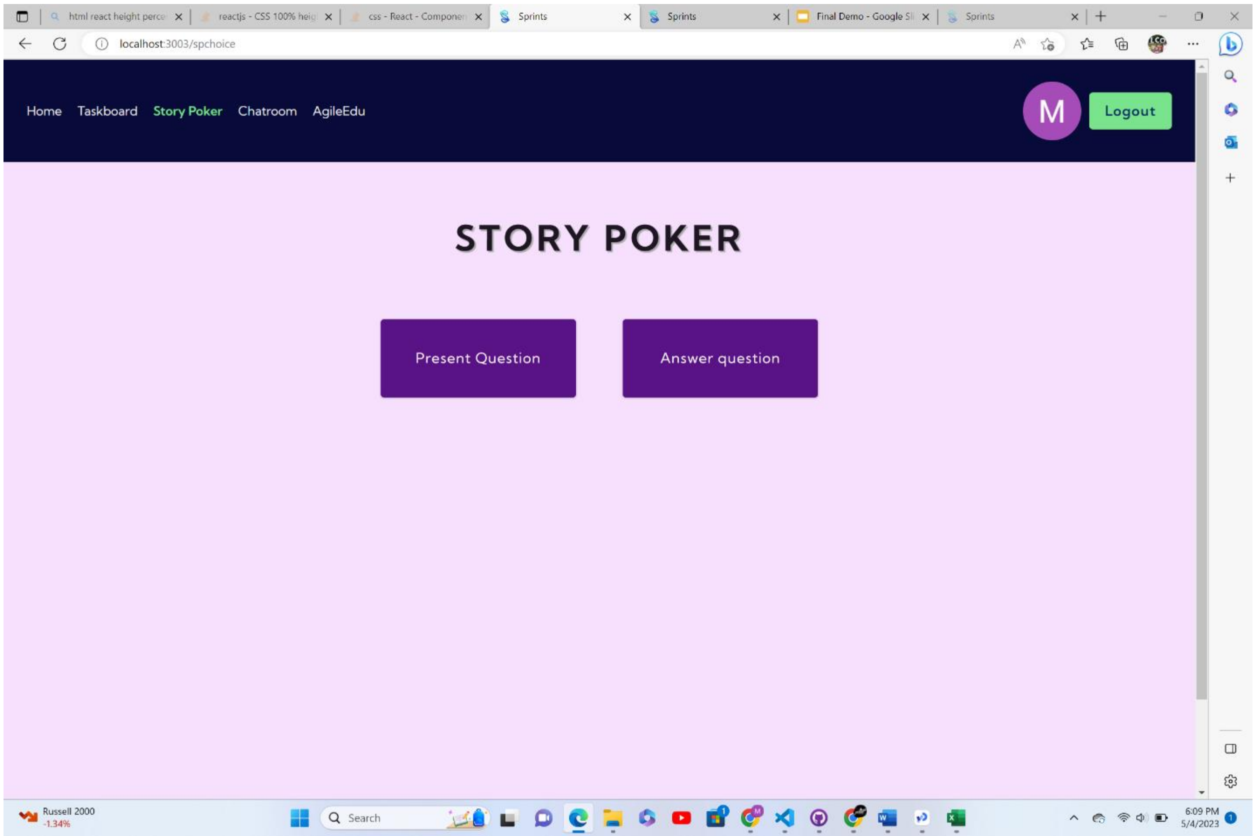
Separate multiple users with commas

Add Task

Reload

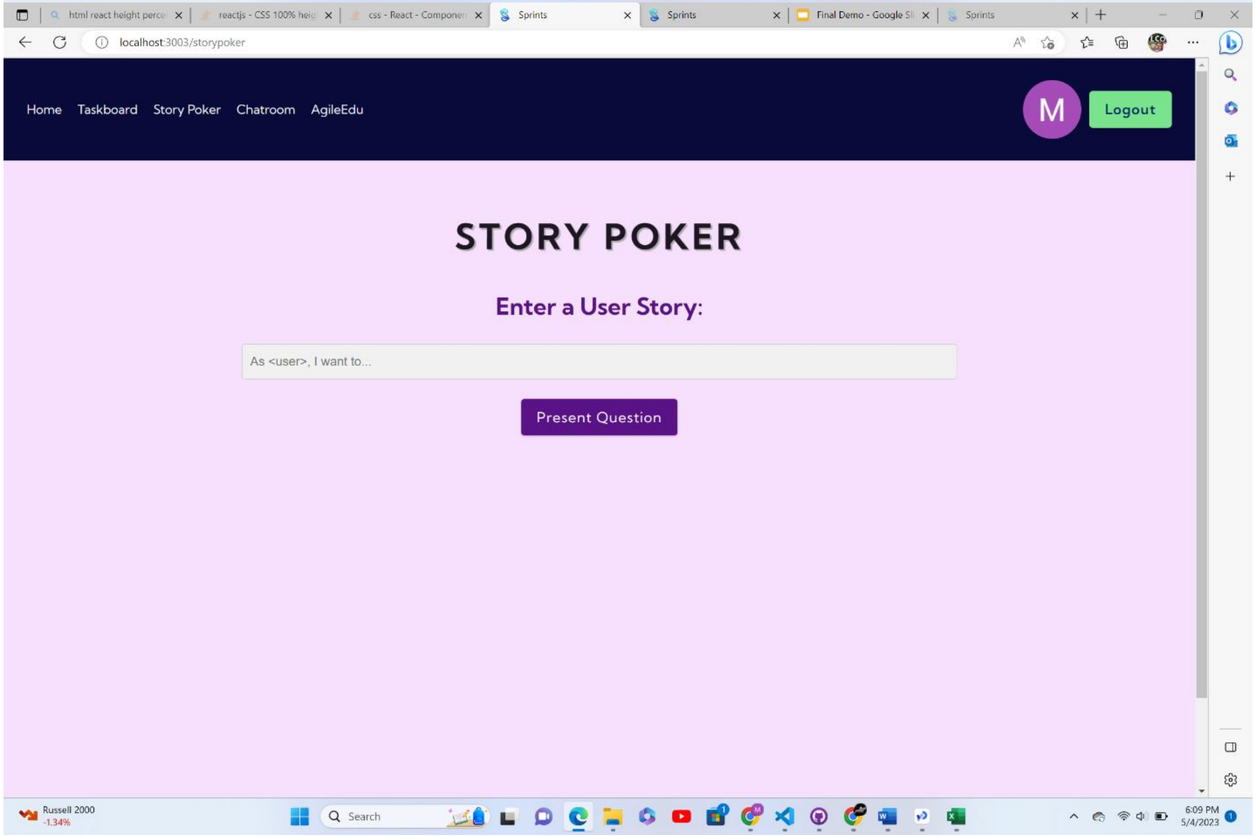
TASKS TABLE

Title ⬍	Priority ⬍	Hours ⬍	Users ⬍	Due Date ⬍	Edit	Delete
Change_Name_testing	medium	2	worker	2023-06-01	✍	Delete
Crossing picnic point on the boat	medium	2	Samyak	2023-06-06	✍	Delete
Ordering tasty snacks for the boat ride	high	1	Manisha	2023-05-10	✍	Delete
Source wood planks for boat	high	3	Casilda	2023-05-06	✍	Delete
Building a boat	medium	21	Allen	2023-05-11	✍	Delete
Go To Gym Every day	medium	2	irving	0001-01-01	✍	Delete



localhost:3003/spchoice

STORY POKER

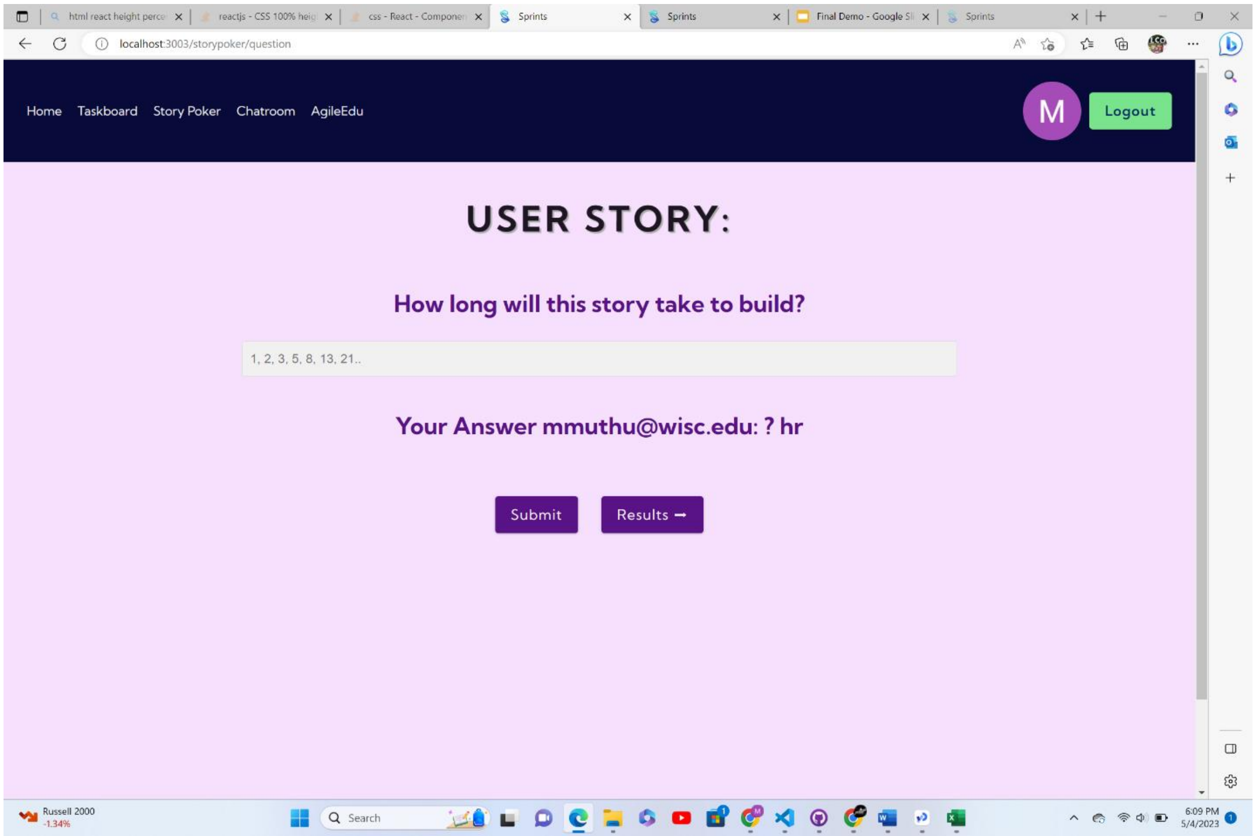


localhost:3003/storypoker

STORY POKER

Enter a User Story:

As <user>, I want to...



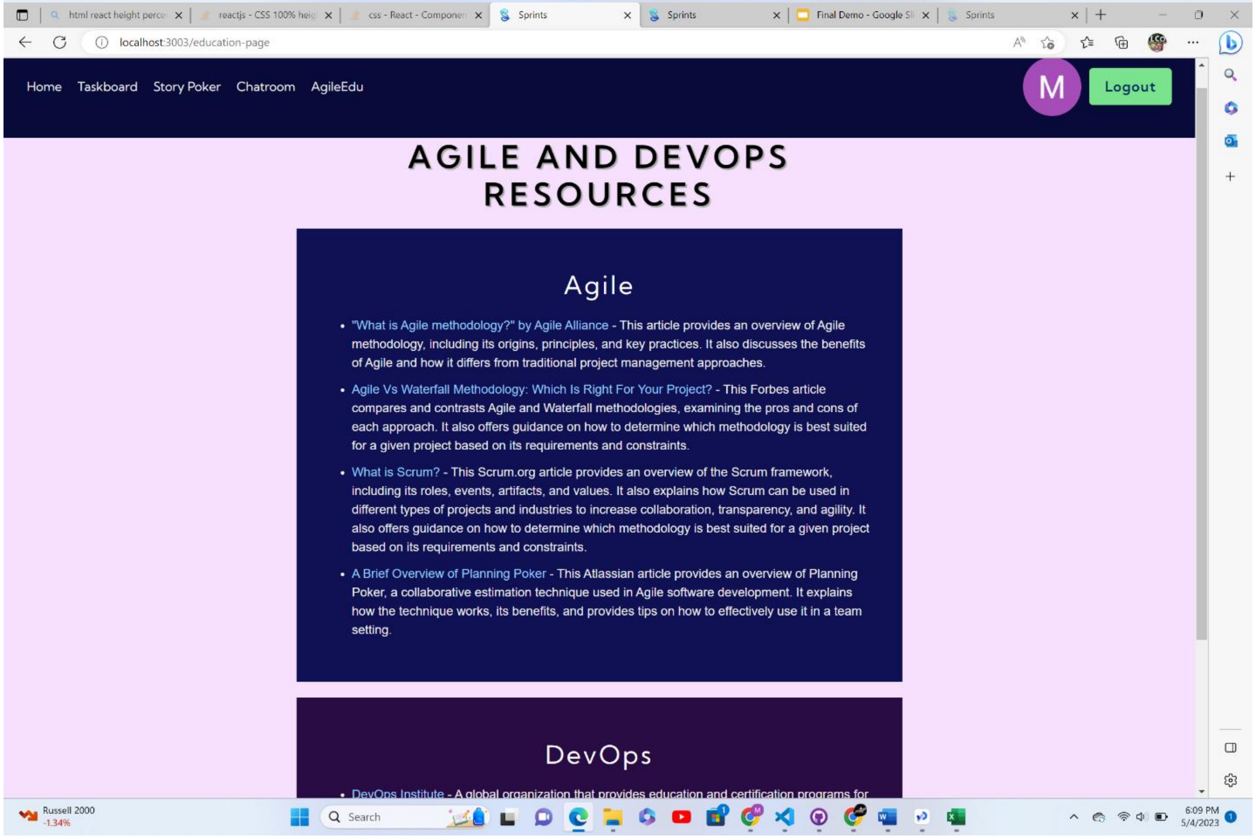
USER STORY:

How long will this story take to build?

1, 2, 3, 5, 8, 13, 21..

Your Answer mmuthu@wisc.edu: ? hr

Submit Results ➞



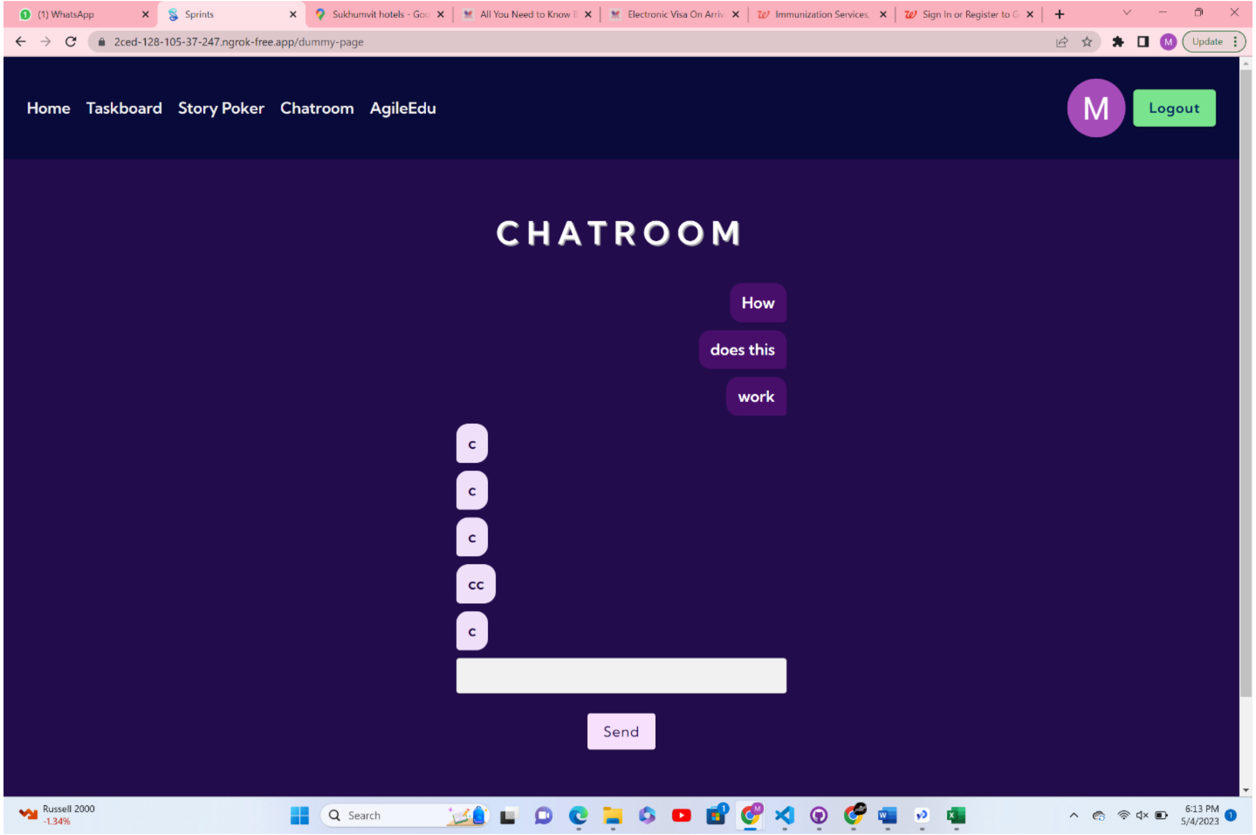
AGILE AND DEVOPS RESOURCES

Agile

- "What is Agile methodology?" by Agile Alliance - This article provides an overview of Agile methodology, including its origins, principles, and key practices. It also discusses the benefits of Agile and how it differs from traditional project management approaches.
- Agile Vs Waterfall Methodology: Which Is Right For Your Project? - This Forbes article compares and contrasts Agile and Waterfall methodologies, examining the pros and cons of each approach. It also offers guidance on how to determine which methodology is best suited for a given project based on its requirements and constraints.
- What is Scrum? - This Scrum.org article provides an overview of the Scrum framework, including its roles, events, artifacts, and values. It also explains how Scrum can be used in different types of projects and industries to increase collaboration, transparency, and agility. It also offers guidance on how to determine which methodology is best suited for a given project based on its requirements and constraints.
- A Brief Overview of Planning Poker - This Atlassian article provides an overview of Planning Poker, a collaborative estimation technique used in Agile software development. It explains how the technique works, its benefits, and provides tips on how to effectively use it in a team setting.

DevOps

- DevOps Institute - A global organization that provides education and certification programs for



How

does this

work

c

c

c

cc

c

Security Requirements

- Authentication and Authorization:
    - The Sprints software platform will require user authentication and authorization for access to the system. Authentication will be managed through a GOOGLE login process).

- Data Privacy and Confidentiality:
    - The Sprints platform will adhere to all relevant data privacy and confidentiality regulations. The system will ensure that all user data is securely stored and transmitted, and that access to sensitive data is restricted to authorized personnel only.

- Logging and Monitoring:
    - The Sprints system will maintain detailed logs of all user activity, system events, and security-related events. These logs will be monitored on a

regular basis to detect and respond to any potential security threats or breaches.

System Requirements

1. Access to the CSL VM (ability to SSH)
2. The web-tool will run on CSL servers and hence require a strong internet connection from users.
3. The CSL server will require at least 1.5GB of RAM for docker containers and 1 GB of local storage on disk.

Database Architecture:

The Sprints system uses a relational database to store all of its data. The database is composed of several tables, including a table for users, projects, sprints, stories, and tasks. Each table is designed to capture the necessary data related to its respective entity. The sql database on the csl server has to have low latency.

Users Table:

The users table contains fields for email address and username. The email is the primary key for this table.

Projects Table:

The projects table contains fields for project ID and project name. The project ID is the primary key for this table.

Team Table:

The team table contains fields for email and project ID. It stores info about who is on what project.

Assigned Table:

The assigned table contains fields for email and task ID. It stores info about who is assigned to what task.

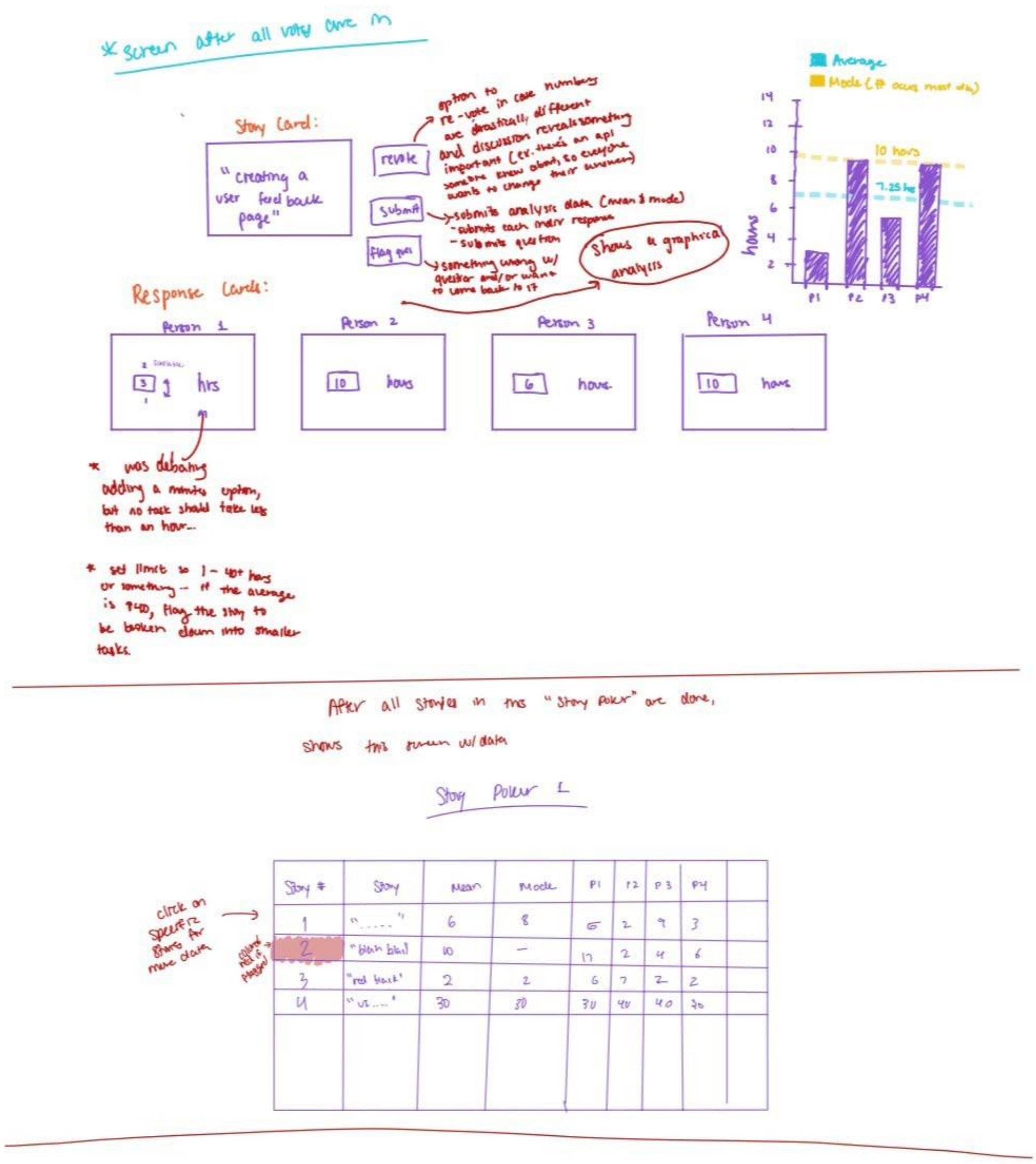
Tasks Table:

The tasks table contains fields for task ID, project ID, task title, due date, hours, priority status, description, sprint num, date created, creator. The task ID is the primary key for this table, and the story ID is a foreign key that references the stories table.

SP Table:

The Story Poker table contains the sender’s email, the answer or question the person posted, and the time of their request.

Story Poker Design Outline:



option to re-vote in case numbers are drastically different and discussion reveals something important (ex. there's an api someone knew about, so everyone wants to change their answer)

- submits analysis data (mean & mode)
- submits each indiv response
- submits question

something wrong w/ question and/or want to come back to it

Shows a graphical analysis

* was debating adding a minutes option, but no task should take less than an hour...

* set limit to 1 - 40+ hrs or something -- if the average is ≥40, flag the story to be broken down into smaller tasks.

After all stories in this "Story Poker" are done, shows this screen w/ data

Story Poker 1

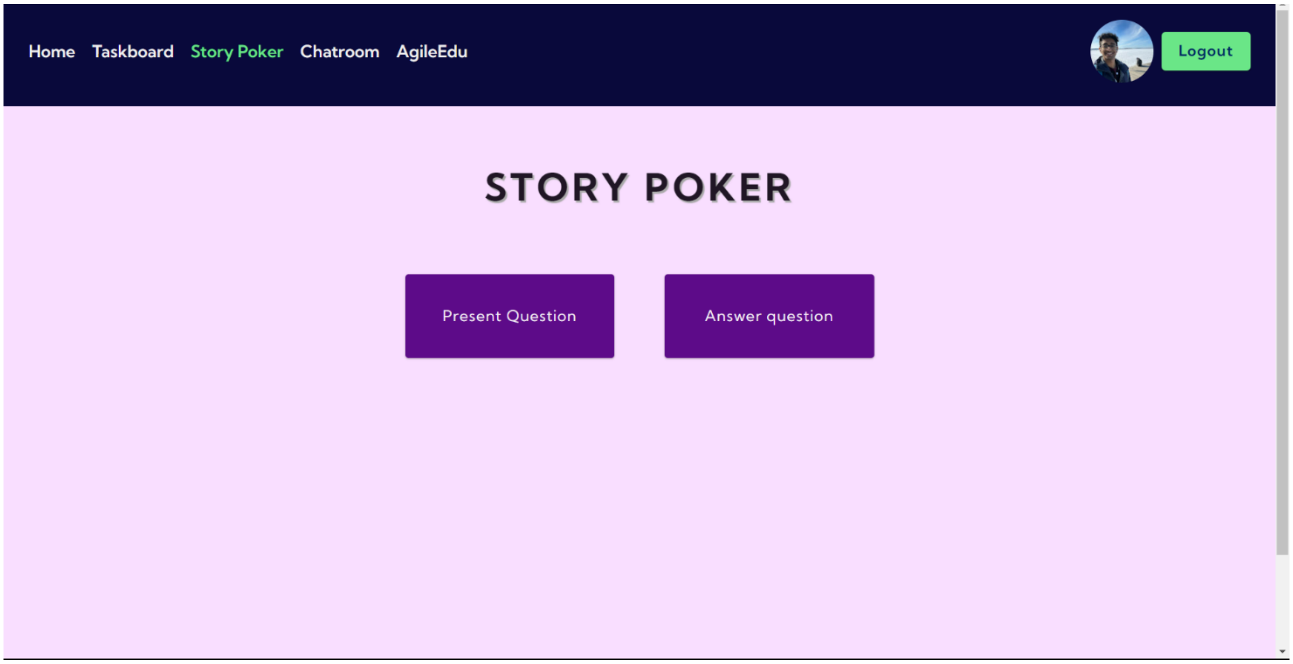
click on specific stories for more data

colored red if flagged

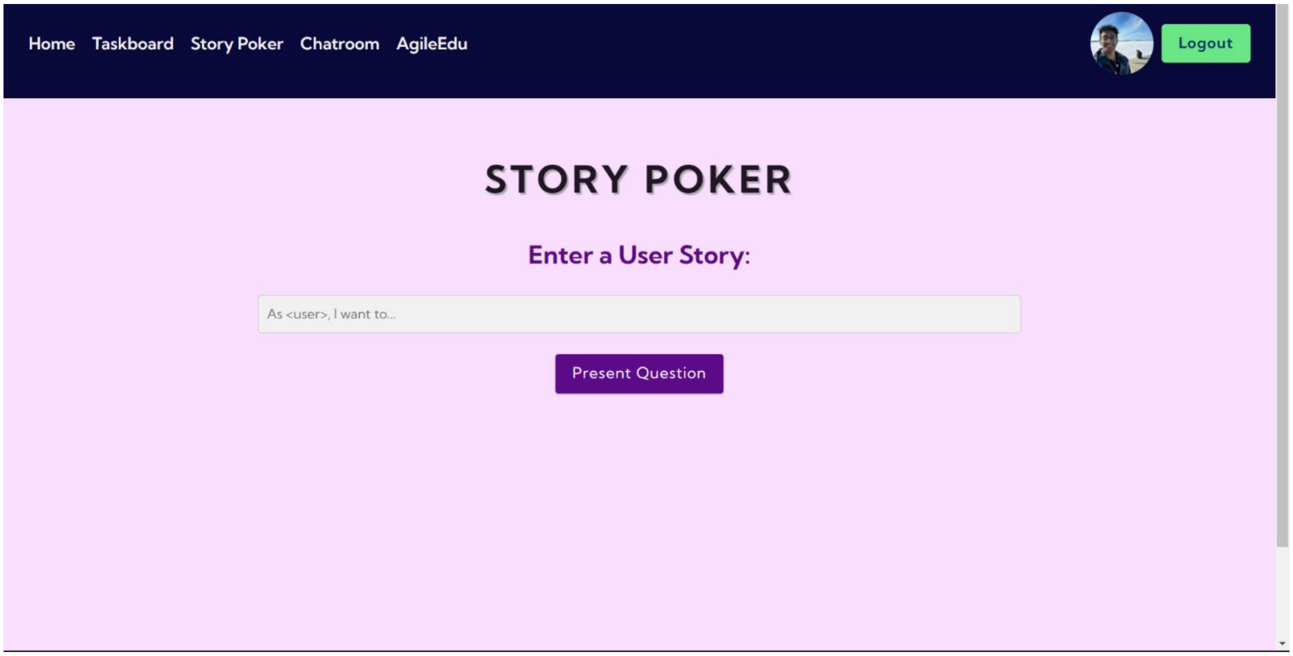
Story #	Story	Mean	Mode	P1	P2	P3	P4	
1	"......"	6	8	6	2	9	3	
2	"blah blah"	10	—	17	2	4	6	
3	"red black"	2	2	6	7	2	2	
4	"us...."	30	30	30	40	40	30	

Final version of story poker page

#1

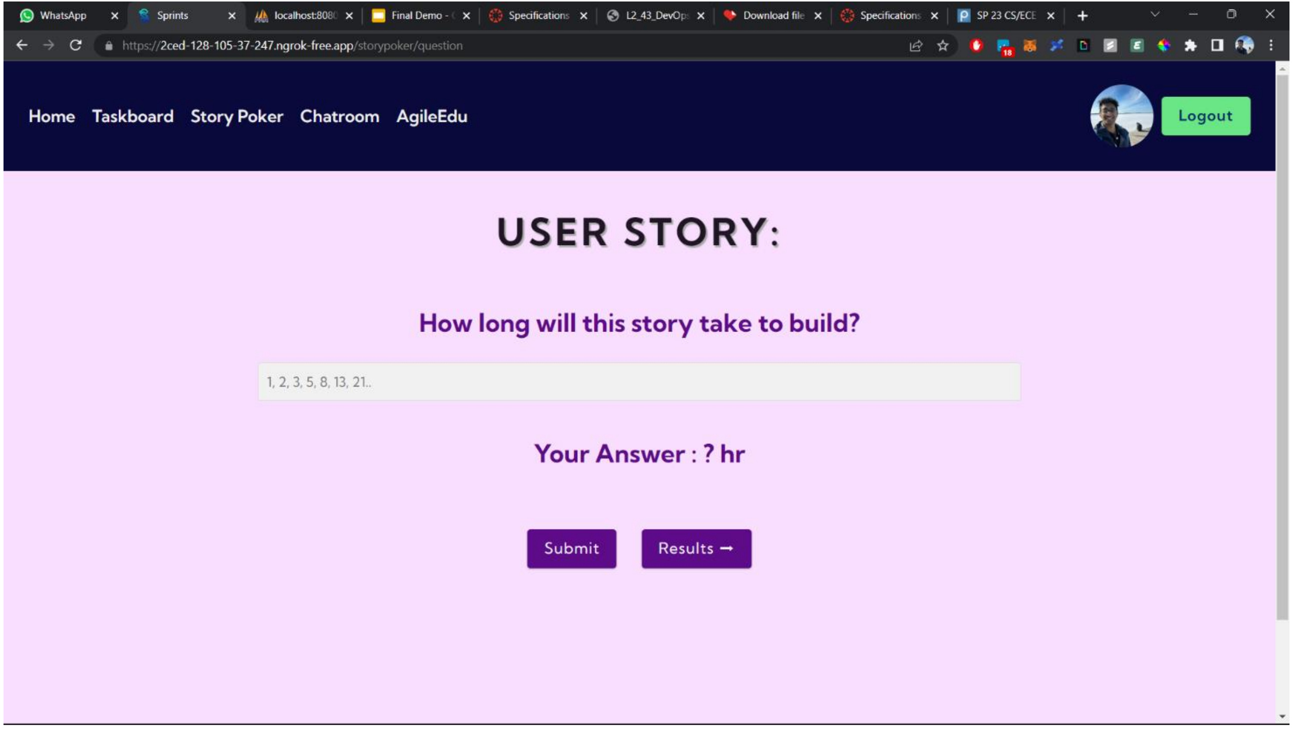


#2 (clicking present question)



#3 (clicking present question)#5, clicking

answer question in #1.



https://2ced-128-105-37-247.ngrok-free.app/storypoker/question

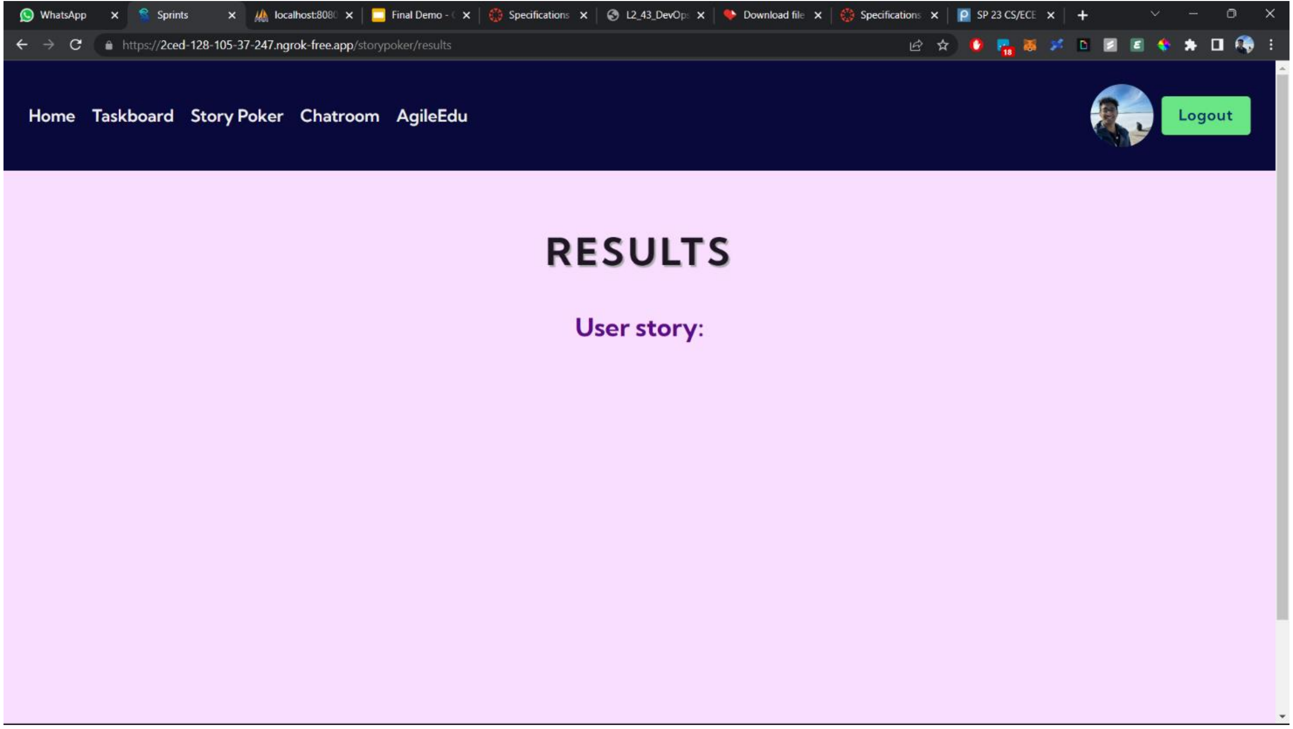
USER STORY:

How long will this story take to build?

1, 2, 3, 5, 8, 13, 21..

Your Answer : ? hr

#4 (clicking Result after submit)



https://2ced-128-105-37-247.ngrok-free.app/storypoker/results

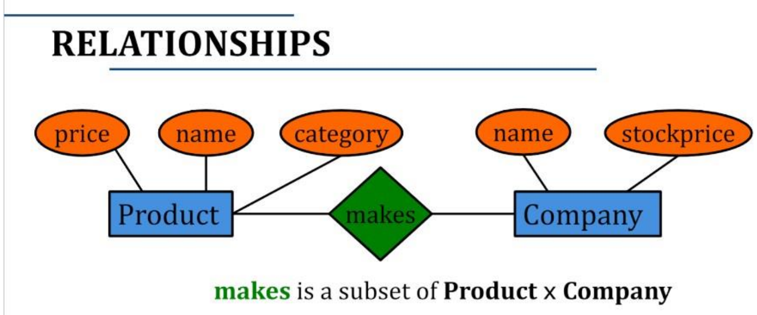
RESULTS

User story:

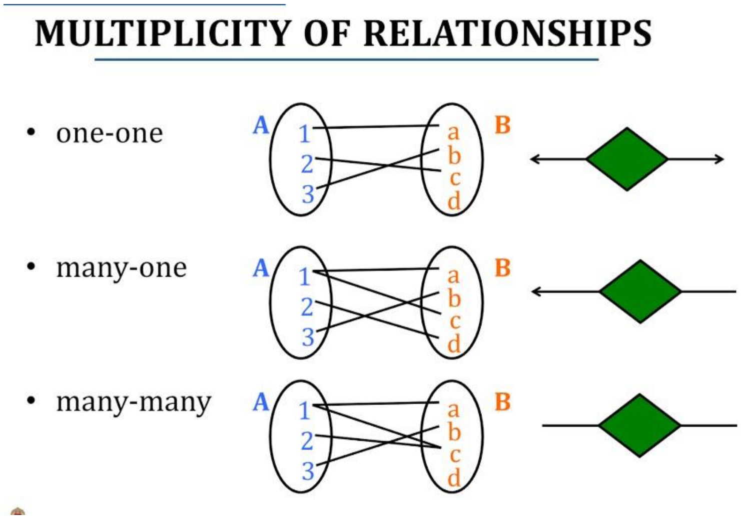
Backend/Databases:

ER Model

Quick Explanation of ER models


RELATIONSHIPS

makes is a subset of **Product** x **Company**


MULTIPLICITY OF RELATIONSHIPS

- one-one
- many-one
- many-many

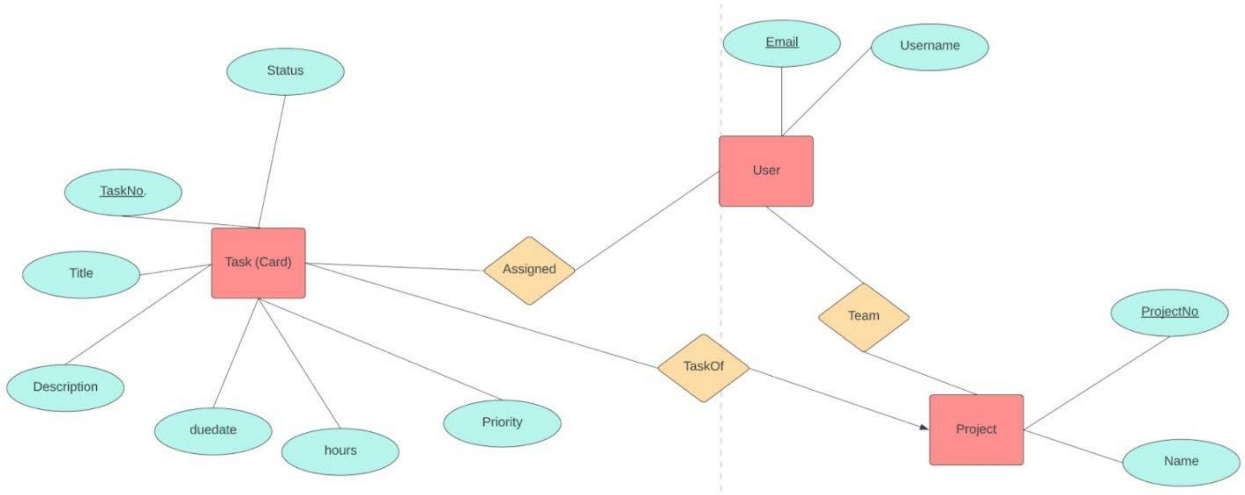
When sending queries to add new rows (7-11), remember that everything that is not null in theschema (above) HAS to be sent in as args.

I’ve added a non exhaustive list of checks to some of the queries. DO THESE CHECKSBEFORE SENDING IN DATA.

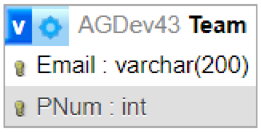
Some of the queries implemented:

1. Get all tasks in a project (different because cards demand a subset of all the taskattributes)
2. Get all users on a task
3. Get all users in a project
4. Get all tasks in a project by status
5. Create new task
   Check if pnum exists before creating a task
6. Create new user
7. Assign user to task
   Stop users from assigning themselves to tasks they’re not on the team forCheck if users exist before assigning
8. Create new project (this is done by assigning a name to the project)
9. Assign users to the project (added to the “team” table)Check if users exist before adding to team

Model of our database

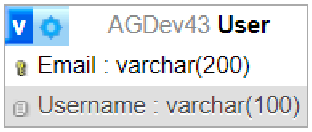


Tables


AGDev43 **Team**

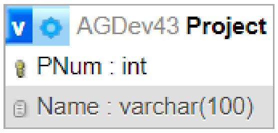
Email : varchar(200)

PNum : int


AGDev43 **User**

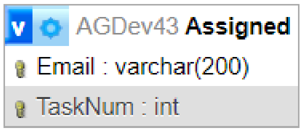
Email : varchar(200)

Username : varchar(100)


AGDev43 **Project**

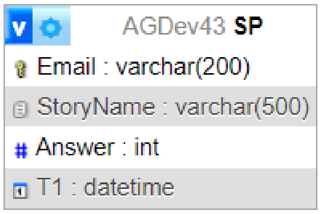
PNum : int

Name : varchar(100)


AGDev43 **Assigned**

Email : varchar(200)

TaskNum : int

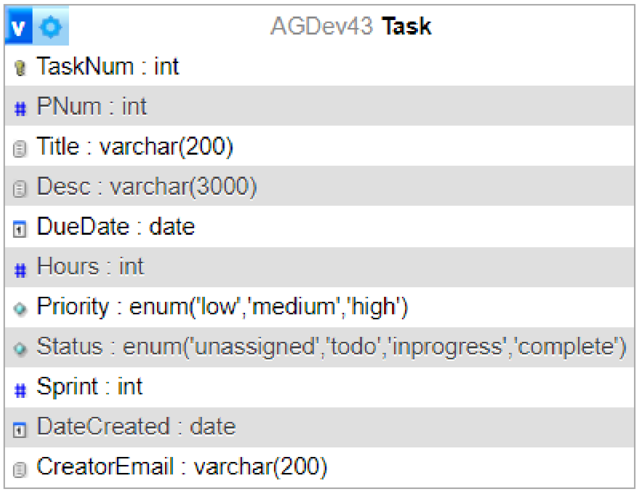

AGDev43 **SP**

Email : varchar(200)

StoryName : varchar(500)

Answer : int

T1 : datetime


AGDev43 **Task**

TaskNum : int

PNum : int

Title : varchar(200)

Desc : varchar(3000)

DueDate : date

Hours : int

Priority : enum('low','medium','high')

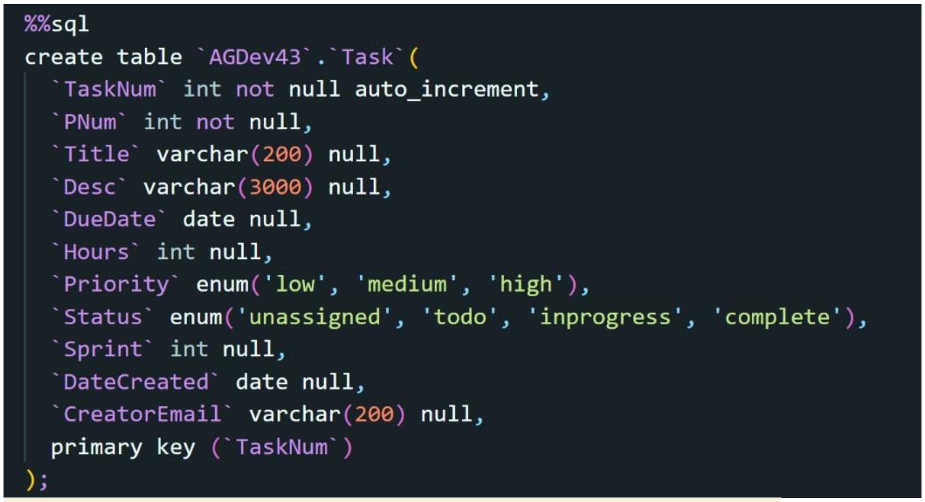
Status : enum('unassigned','todo','inprogress','complete')

Sprint : int

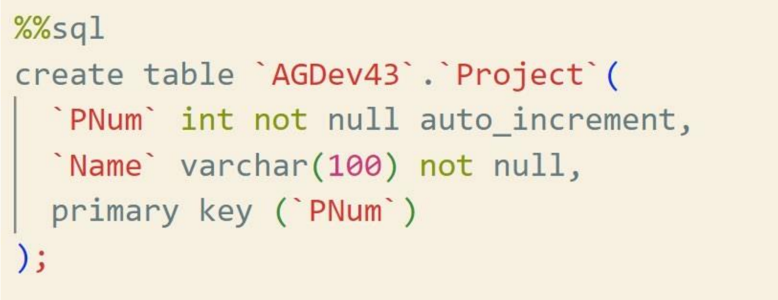
DateCreated : date

CreatorEmail : varchar(200)

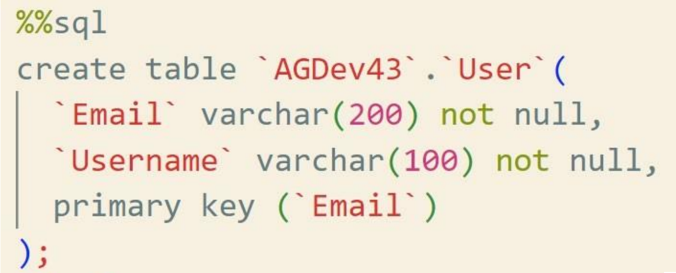
Schema



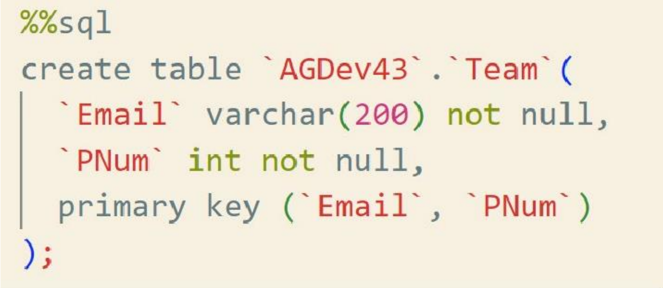
```
%%sql
create table `AGDev43`.`Task`(
  `TaskNum` int not null auto_increment,
  `PNum` int not null,
  `Title` varchar(200) null,
  `Desc` varchar(3000) null,
  `DueDate` date null,
  `Hours` int null,
  `Priority` enum('low', 'medium', 'high'),
  `Status` enum('unassigned', 'todo', 'inprogress', 'complete'),
  `Sprint` int null,
  `DateCreated` date null,
  `CreatorEmail` varchar(200) null,
  primary key (`TaskNum`)
);
```



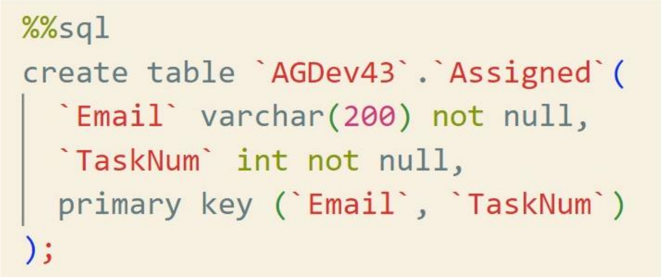
```
%%sql
create table `AGDev43`.`Project`(
  `PNum` int not null auto_increment,
  `Name` varchar(100) not null,
  primary key (`PNum`)
);
```



```
%%sql
create table `AGDev43`.`User`(
  `Email` varchar(200) not null,
  `Username` varchar(100) not null,
  primary key (`Email`)
);
```



```
%%sql
create table `AGDev43`.`Team`(
  `Email` varchar(200) not null,
  `PNum` int not null,
  primary key (`Email`, `PNum`)
);
```



```
%%sql
create table `AGDev43`.`Assigned`(
  `Email` varchar(200) not null,
  `TaskNum` int not null,
  primary key (`Email`, `TaskNum`)
);
```