23/25
Sprint1 should implement these functions: user management, board object and visualization, and piece placement for both players. Your project has partially met the requirements. (-1.5)
The acceptance criteria need to be improved to capture the specific rules of the game and the features of the computer opponent (-0.5)

Team Project Sprint #1 Instructions

Please read the instructions carefully. All members of your team should discuss the instructions together to ensure that everyone is on the same page.

Objectives

Create a brief project description, specify all software requirements as user stories and acceptance criteria, and implement the primitive functions (i.e., user management, board object and visualization, and piece placement for both players). Each team should meet at least once a week.

Deliverable and Grading Policy

1. Project Report (**15 points**)

   The project report should include the following sections:

   I. Project description (micro-charter), which should result from group discussion **(1 point)**.

   II. User stories using the template discussed in class. **(2 points)**
   Provide a complete list of user stories and estimated efforts for the target software.

   III. Acceptance criteria using the template discussed in class. **(8 points)**
   Provide complete acceptance criteria for each of the user stories. Note that, although some of the user stories will be implemented in the future sprints, their acceptance criteria need to be defined in the first sprint. You may continue to improve the user stories and acceptance criteria in the next sprint.

   IV. Implementation tasks **(2 points)**
   Describe the production code, automated test code or manual test case for each user story and acceptance criterion related to the implementation of the primitive functions, including **user registration**, **login**, **logout**, **board visualization**, and **piece placement**. For each acceptance criterion of every user story for the primitive functions, you need to implement at least one test (either test code or manual test case).

   V. Minutes of ALL meetings, including, but not limited to: project/sprint planning meeting, stand-up meeting, backlog grooming, retrospective meeting, and pair programming (or development) session. **(2 points)**

   VI. A table of buddy ratings. Individual members may email their buddy ratings to the instructor or teaching assistant.

   Each team only needs to submit one report. For an individual member to receive the credit for this part of the project, the team's project report must include explicit evidence of his/her contribution (e.g., his/her name is listed as a developer).

PROJECT REPORT SPRING 1

MICRO CHARTER

Author: Bayard Rucker

Project name: delta

This project is to build a web app using an OOP language that allows users to play American checkers against each other. Our development team vision is to deliver a secure and scalable checkers game while using the agile development mythology and multiple modern technologies such as mySQL and the Django web framework.

The purpose of this project is to deliver to our product owners a well designed program and deploy a web app for users to play American checkers. Our aim is a stable application built on a modern framework allowing the dev team to learn new technologies and hone necessary soft skills that are imperative for working on dev teams and in large scale projects.

The business value for our product owners is in allowing users to play checkers as well as the flexible design of the game that will allow for new functionality and additional games to be added. The primary product owner for the project is our professor Dianxiang Xu however all other students in class are encouraged to try the app as well.

The dev team's goal is to use agile for our development mythology. There are many different ways to evaluate a team's value such as tracking K locks or the number of stories completed. The team focus is on delivery of high quality working software, getting the local Django framework running and then integrating with our mysql backend as well as getting the logic for the game engine finished. The team's biggest hurdles heading into the next sprint are a lack of testing. Like many projects, certain decisions were made with the understanding that some amount of refactoring would be needed so the team needs to be ready to integrate and refactor as well as robustly test our code. The second major hurdle is in connection our frontend Django ul and back end game engine however, given the robust documentation from Django, this should be a more straightforward challenge to overcome.

TEAM PROJECT SPRINT #1

REPORT TEMPLATE

Team Name: Delta
Team Members: Bayard Rucker, Muhammad Usman, Zeal Patel, Ergin Bostanci, Sabrina Djeddi

I. Project Micro-Charter (no more than one page)

Provide a brief description about the project, including the following elements:

- Project name
- Vision statement: describe the future that you are trying to create
- Mission statement /project purpose
- Elevator pitch: no more than a few sentences
- Business value
- Customers and users: people who will make buying decisions or actually interact with your product
- Metrics: how to measure the business value
- Milestones: important points in time
- Risks: things that may threaten or derail your project
- Authors of this micro-charter

II. User Stories

NOTE: stories are pointed in terms of complexity. They are meant to be a general marker for the average developers skills of the team. While they are not a one to one mapping of time spent developing a general rule of thumb is that a 3 point story is about one fully day of development work. Given that most students don't work full 8 hour shifts like industry. It's safe to assume a 3 point story might take a little longer in terms of development time.

ID	User Story Name	User Story Description	Priority	Estimated effort (hours)	Actual effort (if completed)	Status (completed, toDo, inProgress)	Developer names
1	Flask and Django spike	As a developer I want to know if Flask or Django will be a better fit for this project	Research	3 points	Points 3	Done	Bayard
2	MYSQL	As a developer I want this project to have a MYSQL back end	Set up DB	Points 3	Points 3	Done	Sabrina
3	set up mvp for app back end	As a developer I want to have a Django back end running	Set up	Points 3	Points3	Done	Bayard
4	Registration/Login pages views and logic	As a user I want a page I can log into or set up a new account with	Front end and db	Points 8	Points 13	Done	Sabrina

5	VUE research spike	Research spike	Re-search	No points	No points	Done	Zeal
6	Look into Django Views research spike	Research spike	Re-search	No points	No points	Done	Zeal & Sabrina
7	build board class	As a user I want a back-end board class	OOD class	Points 3	Points 3	Done	Bayard &Usman
8	build game class	As a user I want a back-end game engine	OOD class	Points13	points13	Done	Usman
9	evaluate com-munication channels	Research spike	Re-search	No points	No points	Done	Zeal
10	build piece class	As a user I want a back-end pieces class	OOD class	Points 3	Points 3	Done	Usman
11	transfer proto-type into Django	As a user I want the proto-type UI transfer to the Django app	Trans-fer	Points 5		In pro-gress	Zeal
12	build player class	As a developer I want to have a class to track active players	OOD design	Points 3		In pro-gress	Ergin
13	add testing li-brary	As a developer I want a cen-tralized testing library	Testing	Points 2	Points 2	Done	Bayard
14	connect board to front end UI	As a user I want a UI page on which the game will be played	UI	Points 3	Points 3	Done	Bayard
15	change Django view functions to classes	Change view functions into classes	Refac-tor	Points 3		In pro-gress	Sabrina
16	add rules page	As I user I want a page where I can see the game rules	UI	Points 3		In pro-gress	Bayard
17	Player stats	As a user I want to see player statistics	UI	Points 3		To do	

III. Acceptance Criteria (AC)

User Story ID and Name	AC ID	Description of Acceptance Criterion	Status (completed, toDo, in Progress)	Developer Names
1 Flask and Django spike	1.1	Given the two different frameworks we are looking into. When a dev has Research them and then presents the finding to the team	Done	Bayard
S:2 MYSQL	2.1	Given a mySQL db when the app starts up then it should Connect MYSQL to Django app	Done	Sabrina
S:3 set up mvp for app back end	3.1	Given a framework has been selected for S:1 when a dev starts working on the app then a base django app should be running	Done	Bayard
S:4 registration/Login pages views and logic	4.1	**Given** a user when a user arrives at the app **then** the user should be able to sign up or sign in.	Done	Sabrina
	4.2	**Given** a user when a user arrives at the app **then** the user is redirected to the login page		
	4.3	**Given** a new user, **when** the user doesn t have a set up account, **then** he has to register(sign up) inorder to create one and access his playing space	Done	Sabrina
	4.4	**Given** a non existent valid user name, **When** the user sign up with this user name, a valid and confirmed password, and a valid email address, **Then** the new player has beencreated and added to the db successfully	Done	Sabrina
	4.5	**Given** a Registration page renders. **When** a user registered successfully **then** they should be redirected to the login page.	Done	Sabrina
	4.6	**Given** a user uses the Registration page when the account has been created for this user **then** it should connects to DB and Django app and save this user into Django-users table and django send a signal to the player table to communicate the new user informations to the player table and this user will become player	Done	Sabrina
	4.7	**Given** a username that already exists, **When** the user try to register with this existent username, a valid and confirmed password, and a valid email address, **Then** the new player account is not created	Done	Sabrina

	4.8	**Given** an non existent valid username, **When** a user try to register with this username, an invalid password, same confirmed password, and a valid email address... **Then** the new account is not created	Done	Sabrina
	4.9	**Given** an non-existent valid username, **When** a user creates an account with this username, a valid email address, a valid password, but different with a confirmed password, **Then** the new player account is not created	Done	Sabrina
	4.10	**Given** an non-existent valid username, **When** a user creates an account with this username, an invalid email address, a valid password, same confirmed password, **Then** the new player account is not created	Done	Sabrina
	4.11	**Given** empty input, **When** a user register with any empty input **Then** the new player account is not created and a message to fill that field will be displayed	Done	Sabrina
	4.12	**Given** the new player **when** this player rendered to the login page and the input credentials are verified with the DB **then** he will be able to access his account	Done	Sabrina
	4.13	**Given** an empty user name and password, **When** a player try to login, the player is redirected to the login page(can not access to his space)	Done	Sabrina
	4.14	**Given** a valid user name and an invalid password, **When** a player try to login, the player is redirected to the login page(can not access to his space)	Done	Sabrina
	4.15	**Given** an invalid username and an invalid password, **When** a player try to login, the player is redirected to the login page(can not access to his space)	Done	Sabrina
	4.16	**Given** an invalid username and a valid password, **When** a player try to login, the player is redirected to the login page(can not access to his space)	Done	Sabrina
	4.17	**Given** a valid username and a valid password, **When** a player try to login, the player is redirected into his session page and can start playing.	Done	Sabrina
S:5 VUE research spike	5.1	Given the different approaches we are looking into. When a dev has Research them and then presents the finding to the team	Done	Zeal
S:6 Look into Django Views research spike	6.1	Given the different approaches we are looking into. When a dev has Research them and then presents the finding to the team	Done	Zeal, Sabrina

S:7 build board class	7.1	Given a new game when board class is called then it should generate a board data structure	Done	Bayard & Usman
	7.2	Given the board class exist then a dev should Add basic read and write when board attributes need to be accessed	Done	Bayard
	7.2	Given a board class then a dev should Add unit test when all the other functions have been built	Done	Bayard
S:8 build game class	8.1	Given a new game then a user should then a user should use the back end game class that rules game logic when playing the game	Done	Usman
	8.2	Given a board class then a dev should Add unit test when all the other functions have been built. Unite testing for this story has been reassigned to story S:18	To do	Usman
S:9 evaluate communication channels	9.1	Given the different approaches we are looking into. When a dev has Research them and then presents the finding to the team	Done	Zeal & Usman
S:10 build piece class	10.1	Given a new game then a user should use the back end pieces class that rules pieces logic when playing the game	To do	None
	10.2	Given the pieces class exist then a dev should Add basic read and write functions when pieces attributes need to be accesses	To do	None
	10.3	Given a pieces class then a dev should Add unit test when all the other functions have been built	To do	None
S:11 transfer prototype into Django	11.3	Given that a lot of UI code has been written then a dev should See what code can be ported over to Django app When it would be more advantages that rewriting it.	Done	Zeal
S:12 build player class	12.1	Given a new player then a user should use the back end pieces class that rules pieces logic when playing the game	In progress	Ergin
	12.2	Given the player class exist then a dev should Add basic read and write functions when player attributes need to be accesses	In progress	Ergin
	12.3	Given a player class then a dev should Add unit test when all the other functions have been built	In progress	Ergin
S:13 add testing library	13.1	Given a centralized testing library is needed then a dev should Integrate Pytest with Django when the .venv in running	Done	Bayard

S:14 connect board to front end UI	14.1	Given that multiple page now render then a user should be able to navigate to the game page when they have logged in	Done	Bayard
	15.1	Given a user in on the game page then the user should see a black page has loaded when they navigate to the page	Done	Bayard
S:15 change Django view functions to classes	15.1	Given that some view functions are not classes then a dev should rewrite them when the are not classes	In progress	Sabrina
	15.1	Given that multiple page now render then a user should be able to navigate to all page when they have logged in	In progress	Sabrina
S:16 add rules page	16.1	Given a user is logged in then the user should see the game rules when the navigate to the rules page	In progress	Bayard
S:17 add player states page	17.1	Given a user is logged in then the user should see there player stats when the navigate to the stats page	To do	None
S:18 unit test-ing for game class	18.1	Write unit test for game class this story is downstream from s:8 Given a board class then a dev should Add unit test when all the other functions have been built.	To do	Usman

IV. Implementation Tasks

Summary of production code

User Story ID and Name	**AC ID**	**Class Name(s)**	**Method Name(s)**	**Developer Name(s)**	**Status**	**Notes (op-tional)**
S:3 set up MVP for app back end	3.1	Code generated in app set up	Code generated in app set up	Bayard	Done	
S:2 MYSQL	2.1	DB connects to app		Sabrina	Done	
S:4 registra-tion page view and logic	4.1	HTML page		Sabrina	Done	
	4.2	Used Django models		Sabrina	Done	
	4.3	Html page		Sabrina	Done	

	4.4	Used dragon models		Sabrina	Done	
S:7 build board class	7.1	Board class	init() Generate_board() get_spaces() space_swap()	Bayard	Done	
	7.2	Board class	init() Generate_board() get_spaces() space_swap()	Bayard	Done	
	7.3		test_board_build() test_board_generate() test_board_get_spaces	Bayard	Done	
S:8 build game class	8.1	Moving errors State	ner() move() Errors did_end() simple_move() jump_avaialbe() farther() pieces_after_sim-ple_moce() Piece_after_jump()	Usman	Done	This story has multiple class and functions and has stories for refactor and adding unit testing
S:9 evaluate communica-tion channels	9.1			Zeal	Done	
S:13 add test-ing library	13.1	Integration with Django running Pytest in .venv run successfully		Bayard	Done	
S:14 connect board to front end UI	14.1	Html page renders		Bayard	Done	

Summary of automated test code (directly corresponding to some acceptance criteria)

User Story ID and Name	Acceptance Criterion ID	Class Name (s) of the Test Code	Method Name(s) of the Test Code	Description of the Test Case (input & expected output)	Status	Developer Name(s)
S:7 build board class	7.1	No need for class test yet	test_board_build test_board_game test_board_getspaces	Simple unit test, check data type, input files, and object length	Done	Bayard Rucker
S:18 unit testing for game class	18.1	Pytest does not need a class	To do	To do	To do	Usman

Summary of manual test cases (directly corresponding to some acceptance criteria)

User Story ID and Name	Acceptance Criterion ID	Test Case Input	Test Oracle (Expected Output)	Status	Notes	Developer Name(s)
S:3 set up MVP for app back end	3.1	App runs in dev env		Done		Bayard
S:2 MYSQL	2.1	Check read and writes in DB		Done		Sabrina
S:4 registration page view and logic	4.1	Page renders		Done		Sabrina
S:8 build game class	8.1	Game is playable from command line		Done		Usman
S:14 connect board to front end UI	14.1	Page renders		Done		Bayard

Summary of other automated or manual tests (not corresponding to the acceptance criteria)

Number	Test Input	Expected Result	Class Name of the Test Code	Method Name of the Test Code	Status	Developer Name(s)

V. Meeting Minutes

Report the minutes of all meetings, including, but not limited to: project/sprint planning meeting, stand-up meeting, backlog grooming, retrospective meeting, and pair programming session.

Date	Time and Duration	Place	Participant Names	Purpose of the Meeting	Specific Action Items
8/27	45 min	Zoom	Bayard, Zeal, Sabrina	Team formation	Set up base team. Talked about possible approaches and high-level details like languages
8/27	45 min	In person	Bayard, Ergin	Team formation	Set up base team. Talked about possible approaches and high-level details like languages
8/30	1.5 hours	Zoom	Full team	First team meeting	Introductions, talked about high-level project requirements, set up Trello board and added initial stories, set up GIT repo
9/6	1 hour	Zoom	Full team	Week meeting time	Checked in on everyone progress. Added a few stories. Decided to use Django as main framework and MYSQL as DB
9/13	2 hours	Zoom	Full team	Week meeting time	Weekly meeting decided not to use VUE for the front end. Base app set up
9/20	1.5 hours	Zoom	Full team	Week meeting time	To be more agile we followed agile practices listed in meeting agenda doc added testing library, walked thru UI updates and how Django connects to MYSQL

9/20	1 hour	In person student union	Bayard, Ergin	Paired programming	Talking about project overview and worked on local set up for MYSQL and setting up player class
9/27	2 hours	Zoom	Full team	Week meeting time	Followed agenda doc. Talked about implemented stories and board class focused on OOP and unit testing
10/4	1 hour	Zoom	Full team	Week meeting time	Focused on updating routing and UI. Added and pointed new stories.
10/8	45 min	Zoom	Bayard, Usman	Code review	Review of game class and talk about next steps

VI. Buddy Ratings

If you don't feel comfortable to include your ratings in this report, you may email your ratings to the instructor or grader.

		Rating receiver				
Rating giver		Bayard Rucker	Muhammad Usman	Zeal Patel	Ergin Bostanci	Sabrina Djeddi
	Bayard Rucker	X	1	1	1	1
	Muhammad Usman	1	X	1	1	1
	zeal Patel	1	1	X	1	1
	Ergin Bostanci	1	1	1	X	1
	Sabrina Djeddi	1	1	1	1	X
	Average	1	1	1	1	1