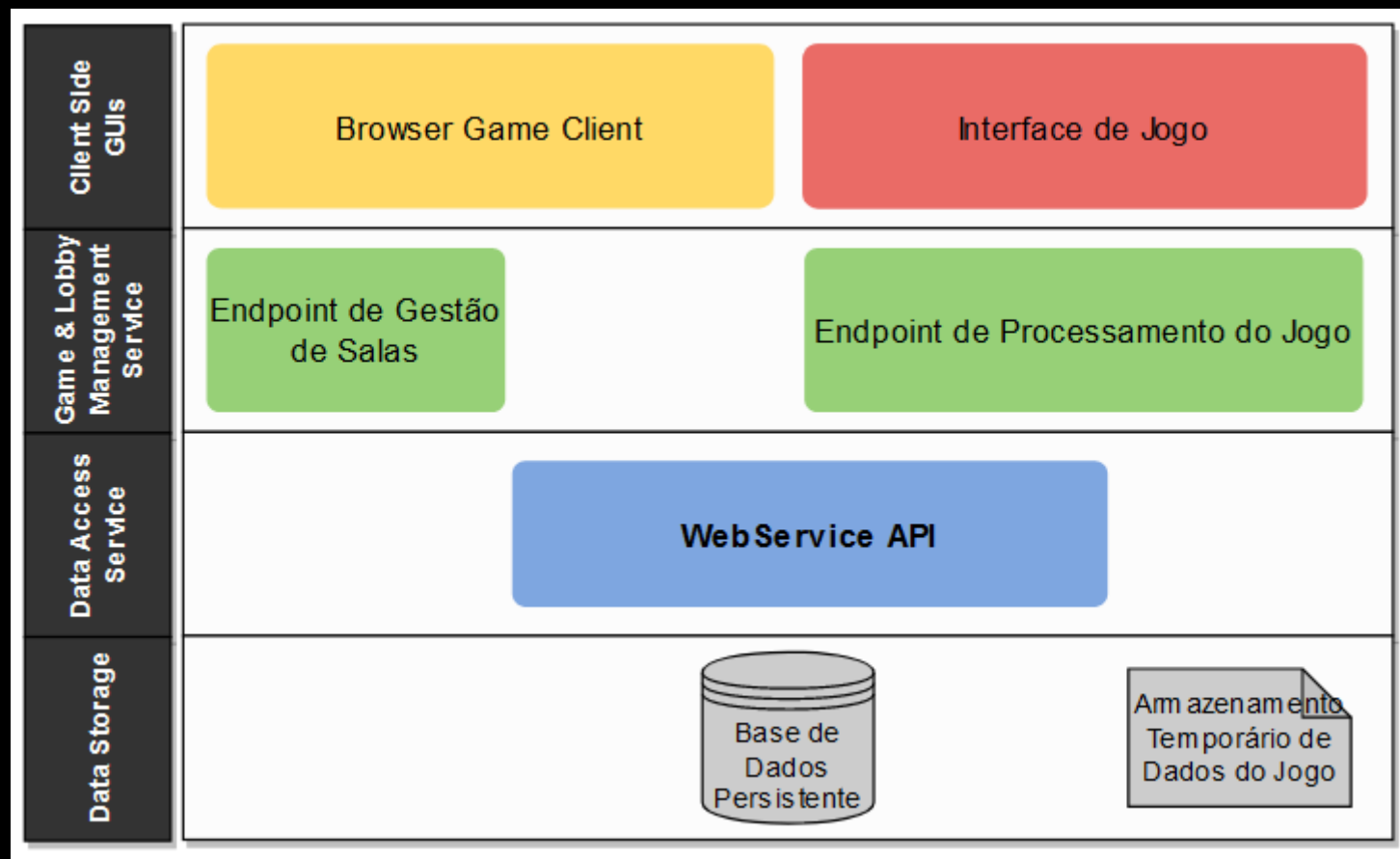


D A R K E N I N G
A G E

MILESTONE 2

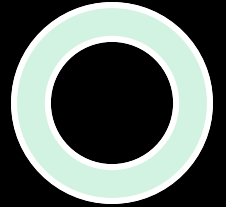


Arquitetura



Sprints do Projeto

- Sprint 1: Desenvolvimento inicial do projeto
 - Lógica de Mapa
 - Cliente de Jogo
 - Comunicação Cliente-Servidor
- Sprint2: Amadurecimento do projeto
 - Gestão de Exércitos, Recursos e Fações
 - Gestão de Salas e Jogadores
 - Lógica de jogo em Frontend



Sprint 2

Nov 25, 2020–Dec 8, 2020

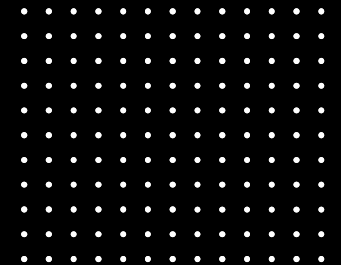
LDS Group 15 / Darkening Age

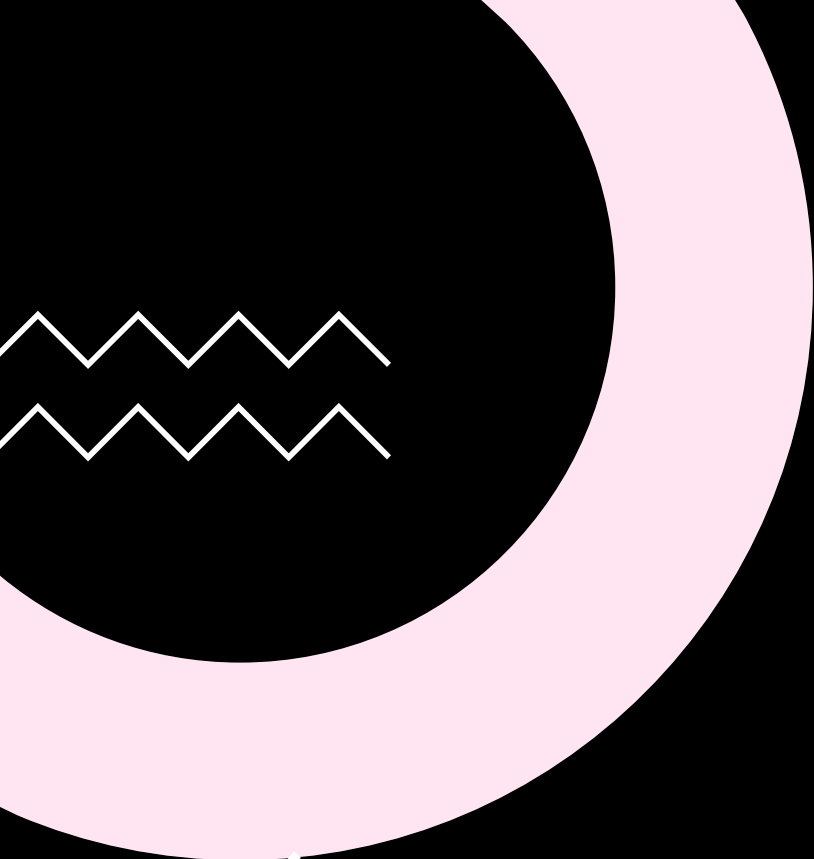
Sprint-1

Nov 7, 2020–Nov 25, 2020

Closed

LDS Group 15 / Darkening Age





Lógica de Jogo em Servidor





Comunicação Cliente Servidor



Websockets: Transmissão de dados via String em formato JSON;



Lógica Cliente-Servidor

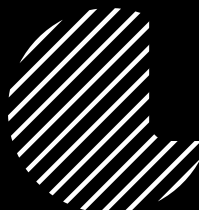
Dois tipos de Cliente: Browser e Unity
Comunicação entre Clientes
unidirecional no início de cada partida



Os Clientes enviam informação para ser processada no Servidor



No final de cada turno, o Servidor envia o novo estado do Jogo



Serviço de mensagens entre jogadores



Gestão de Salas

Dois tipos de sala

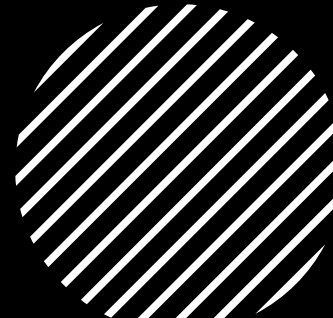
Sala de Lobby e Sala de Jogo

Sala de Lobby original
Sala de Jogo quando o jogo é lançado

Comunicação entre os dois Clientes:
Browser -> Unity

Cada Sala de Jogo possui a sua instância de Jogo

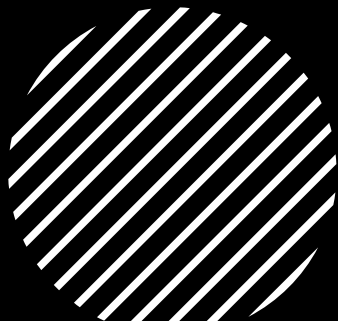
Faço dos Jogadores é definida quando da sua entrada na Sala de Jogo





Gestão de Salas

- Utilização de *Singleton* e gestão de concorrência
- Verificações de integridade lógica e de dados
- Utilização de um Protocolo com regras definidas para comunicação



Protocolo de Comunicação

Descrição	Tipo	Conteúdo
Criar exército	"eventType": 'CreateArmy'	data:[faction, general, name, region, unit]
Criar unidade	"eventType": 'CreateUnit'	data:[faction, army, unit, region]
Reforçar unidade	"eventType": 'ReinforceUnit'	data:[unitID, army]
Mover unidade	"eventType": 'MoveArmy'	data:[army, region, faction]
Trocar posição de unidades	"eventType": 'SwapArmies'	data:[army, region, faction]
Atacar Região	"eventType": 'AttackRegion'	data: [attackerregion, defenderregion, attackarmy, defencearmy]
Anexar região neutra	"eventType": 'AnnexNeutralRegion'	data:[army, _targetRegion, comingRegion]

(Alvo) Descrição	Tipo	Conteúdo
(Jogador que fez <i>host</i>) Receção informação lobby após <i>host</i>	"eventType": 'lobby-alert'	"title": 'new-room-id', "data": 'lobby_room_id'
(Jogador que deu <i>join</i>) Receção informação lobby após <i>join</i>	"eventType": 'lobby-alert'	"title": 'lobby-players', "data": ['player_id', 'player_id']
(Todos) Notificação de novo player na sala	"eventType": 'lobby-alert'	"title": 'player-connected', "data": 'player_id'
(Todos) Notificação da saída de um player	"eventType": 'lobby-alert'	"title": 'player-disconnected', "data": 'player_id'
(Todos) Notificação de atualização do <i>host</i>	"eventType": 'lobby-alert'	"title": 'new-host', "data": 'host_player_id'
(Todos) Notificação da criação da sala de jogo	"eventType": 'lobby-alert'	"title": 'game-started', "data": 'game_room_id'
(Todos) Mensagem de chat e conteúdo	"eventType": 'chat-message'	"user": 'user_id', "msg": 'conteúdo', "time": 'hora_receção'
(Cliente onde ocorreu o erro) Mensagem de erro	"errorType": 'erro'	"message": 'mensagem_de_erro'





Frontend Unity

Integração dos Eventos do Servidor

Comunicação com o servidor

- Particular atenção ao uso de Programação Concorrente

Verificações lógicas com vista à otimização de eventos e envios de mensagem

- Bloqueio de certas ações tanto no servidor como no cliente

Utilização de “Dummies” para representar informação futura



A Equipe e Responsabilidades

SCRUM MASTER / DEVELOPER



José Baltar

- Gestão de comunicação e salas
- Frontend browser do Cliente

PRODUCT OWNER / DEVELOPER



Rodrigo Coelho

- Lógica de Jogo em Servidor
- Protocolo de Comunicação

DEVELOPER



Fábio Mendes

- Lógica de Jogo em Cliente
- Frontend em Unity





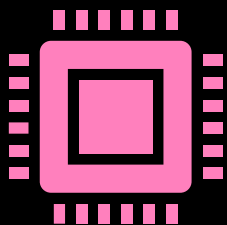
Divergência entre Planeado e Executado

- Matchmaking
- Autenticação
- Rest API

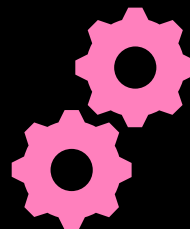




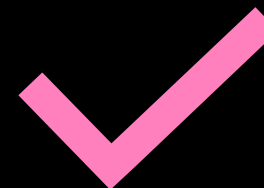
Tarefas Futuras



Conclusão da construção da
lógica de jogo (Cliente e Servidor)



Correções de lógica, otimizações
e melhorias



Implementação do GameClient
em browser e o serviço REST API

