

Requesting and Receiving Session Lists by Role

Code base seminar

Backend.AI

요약

세션

가져오기 & 실행

데이터 & 폴더

통계

관리

사용자

실행 환경

자원

환경 설정

관리

정보

약관 · 개인정보보호 · Backend.AI에 대하여

Lablup Inc. 21.03.11.210913

localhost

환영합니다.Admin Lablup님

현재 프로젝트 default

사용자 Admin Lablup

로그아웃

자원 사용량

CPU

2/2100%2/2100%

RAM

1.00/12.00GB8%1.00/12.00GB8%

세션

1/13%

실행중종료기타

시작

접근키

(최대 길이 64자)

	#	User ID	세션 정보	상태	제어	구성	사용량	예약시간	실행노드
<input type="radio"/>	1	admin@lablup.com	AdminSession testing/repository REPO	RUNNING	<div></div> <div>>_</div> <div></div> <div></div>	default ^{RG} 2코어 _GPU1.00GB Demo	CPU RAM I/O R: 0.0MB / W: 0.4MB	2021. 10. 6. 오후 6:59:13 (00:05:36)	i-Sujinui-MacBookPro.local
<input type="radio"/>	2	user@lablup.com	UserSession testing/repository REPO	RUNNING	<div></div> <div>>_</div> <div></div> <div></div>	default ^{RG} 1코어 _GPU3.00GB Demo	CPU RAM I/O R: 6.3MB / W: 0.4MB	2021. 10. 6. 오후 6:57:30 (00:07:19)	i-Sujinui-MacBookPro.local

Request session list

backend.ai-webui/src/components/backend-ai-session-list.ts

`_viewStateChanged()`



`_refreshJobData()`



`globalThis.backendaiclient.
computeSession.list()`



`this.client.query()`

```
async _viewStateChanged(active) {  
  await this.updateComplete;  
  if (active === false) {  
    return;  
  }  
  // If disconnected  
  if (typeof globalThis.backendaiclient === 'undefined'  
    || globalThis.backendaiclient === null  
    || globalThis.backendaiclient.ready === false) {  
    document.addEventListener('backend-ai-connected', () => {  
      ...  
      this._refreshJobData();  
    }, true);  
  } else {  
    ...  
    this._refreshJobData();  
  }  
}
```

Request session list

backend.ai-webui/src/components/backend-ai-session-list.ts

`_viewStateChanged()`



`_refreshJobData()`



`globalThis.backendaiclient.
computeSession.list()`



`this.client.query()`

```
async _viewStateChanged(active) {  
  await this.updateComplete;  
  if (active === false) {  
    return;  
  }  
  // If disconnected  
  if (typeof globalThis.backendaiclient === 'undefined'  
    || globalThis.backendaiclient === null  
    || globalThis.backendaiclient.ready === false) {  
    document.addEventListener('backend-ai-connected', () => {  
      ...  
      this._refreshJobData();  
    }, true);  
  } else {  
    ...  
    this._refreshJobData();  
  }  
}
```


Request session list

backend.ai-webui/src/components/backend-ai-session-list.ts

_viewStateChanged()

- BackenAiSessionList class 는 BackendAIPage 를 상속
- <BackendAIPage>
attribute 가 변경될 때마다 호출되는
사용자 지정요소 API 콜백 함수
attributeChangedCallback() 에서
_viewStateChanged() 호출



```
export default class BackendAiSessionList extends BackendAIPage
```



backend.ai-webui/src/components/backend-ai-page.ts

```
attributeChangedCallback(name: string, oldval: string|null, newval: string|null): void {  
  if (name == 'active' && newval !== null) {  
    this.active = true;  
    this._viewStateChanged(true);  
  } else if (name === 'active') {  
    this.active = false;  
    this._viewStateChanged(false);  
  }  
  super.attributeChangedCallback(name, oldval, newval);  
}
```

Request session list

backend.ai-webui/src/components/backend-ai-session-list.ts

_viewStateChanged()



_refreshJobData()



globalThis.backendaiclient.
computeSession.list()



this.client.query()

```
async _viewStateChanged(active) {  
  await this.updateComplete;  
  if (active === false) {  
    return;  
  }  
  // If disconnected  
  if (typeof globalThis.backendaiclient === 'undefined'  
    || globalThis.backendaiclient === null  
    || globalThis.backendaiclient.ready === false) {  
    document.addEventListener('backend-ai-connected', () => {  
      ...  
      this._refreshJobData();  
    }, true);  
  } else {  
    ...  
    this._refreshJobData();  
  }  
}
```

Request session list

backend-ai-webui/src/components/backend-ai-session-list.ts

_refresh

- 세션

- RUN

→ 53

- FINIS

→ 30

Backend.AI

요약

세션

가져오기 & 실행

데이터 & 폴더

통계

관리

사용자

실행 환경

자원

환경 설정

관리

정보

약관 · 개인정보보호 · Backend.AI에 대하여

Lablup Inc. 21.03.11.210913

환영합니다.Admin Lablup님

현재 프로젝트 default

사용자 Admin Lablup

로그아웃

자원 사용량

CPU 2/2 100%

RAM 1.00/12.00GB 8%

세션 1/1 3%

실행중

종료

기타

...

시작

접근키

(최대 길이 64자)

#	User ID	세션 정보	상태	제어	구성	사용량	예약시간	실행노드
1	admin@lablup.com	AdminSession testing/repository REPO	RUNNING	<div></div> <div>>_</div> <div></div>	default ^{RG} 2코어 1.00GB GPU Demo	CPU RAM I/O R: 0.0MB / W: 0.4MB	2021. 10. 6. 오후 6:59:13 (00:05:36)	i-Sujinui-MacBookPro.local
2	user@lablup.com	UserSession testing/repository REPO	RUNNING	<div></div> <div>>_</div> <div></div>	default ^{RG} 1코어 3.00GB GPU Demo	CPU RAM I/O R: 6.3MB / W: 0.4MB	2021. 10. 6. 오후 6:57:30 (00:07:19)	i-Sujinui-MacBookPro.local

```
-resource-refreshed' event  
refreshTime to 5000 for running list else
```

```
, 'SCHEDULED', 'PREPARING', 'PULLING'];
```

ED

```
master/src/ai/backend/manager/models/
```

```
, 'SCHEDULED', 'PREPARING', 'PULLING'];
```

```
&& status.includes('SCHEDULED')) {
```

```
tes')) {
```

```
}  
...  
}
```

Request session list

backend.ai-webui/src/components/backend-ai-session-list.ts

_refreshJobData()

- 현재 탭 정보(this.condition)에 맞춰 status 지정

```
/**
 * Refresh the job data - data fields, sessions, etc.
 *
 * @param {boolean} refresh - if true, dispatch the 'backend-ai-resource-refreshed' event
 * @param {boolean} repeat - repeat the job data reading. Set refreshTime to 5000 for running list else
30000
 */
async _refreshJobData(refresh = false, repeat = true) {
  await this.updateComplete;
  if (this.active !== true) {
    return;
  }
  if (this.refreshing === true) {
    return;
  }
  this.refreshing = true;
  this.spinner.show();
  let status: any;
  status = 'RUNNING';
  switch (this.condition) {
    case 'running':
      status = ['RUNNING', 'RESTARTING', 'TERMINATING', 'PENDING', 'SCHEDULED', 'PREPARING', 'PULLING'];
      break;
    case 'finished':
      status = ['TERMINATED', 'CANCELLED']; // TERMINATED, CANCELLED
      break;
    case 'others':
      status = ['TERMINATING', 'ERROR']; // "ERROR", "CANCELLED"..
      // Refer https://github.com/lablup/backend.ai-manager/blob/master/src/ai/backend/manager/models/
kernel.py#L30-L67
      break;
    default:
      status = ['RUNNING', 'RESTARTING', 'TERMINATING', 'PENDING', 'SCHEDULED', 'PREPARING', 'PULLING'];
  }
  if (!globalThis.backendaiclient.supports('avoid-hol-blocking') && status.includes('SCHEDULED')) {
    status = status.filter((e) => e !== 'SCHEDULED');
  }
  if (globalThis.backendaiclient.supports('detailed-session-states')) {
    status = status.join(',');
  }
  ...
}
```


Request session list

backend.ai-webui/src/components/backend-ai-session-list.ts

_refreshJobData()

- 조회할 fields 설정
- superadmin이라면 agent 정보도 조회

```
const fields = [
  'id', 'session_id', 'name', 'image',
  'created_at', 'terminated_at', 'status', 'status_info',
  'service_ports', 'mounts',
  'occupied_slots', 'access_key'
];
if (globalThis.backendaiclient.supports('multi-container')) {
  fields.push('cluster_size');
}
if (globalThis.backendaiclient.supports('multi-node')) {
  fields.push('cluster_mode');
}
if (globalThis.backendaiclient.supports('session-detail-status')) {
  fields.push('status_data');
}
if (this.enableScalingGroup) {
  fields.push('scaling_group');
}
if (this._connectionMode === 'SESSION') {
  fields.push('user_email');
}
if (globalThis.backendaiclient.is_superadmin) {
  fields.push('containers {container_id agent occupied_slots live_stat last_stat}');
} else {
  fields.push('containers {container_id occupied_slots live_stat last_stat}');
}
```

Superadmin

실행중 종료 기타 ... 시작									
접근키 (최대 길이 64자)									
#	User ID	세션 정보	상태	제어	구성	사용량	예약시간	실행노드	
1	admin@lablup.com	AdminSession testing/repository REPO	RUNNING	>_	default ^{RG} 2코어 1.00GB _GPU Demo	CPU RAM I/O R: 0.0MB / W: 0.4MB	2021. 10. 8. 오후 12:17:20 (00:02:11)	i-Sujinui-MacBookPro.local	
2	user@lablup.com	UserSession testing/repository REPO	RUNNING	>_	default ^{RG} 1코어 3.00GB _GPU Demo	CPU RAM I/O R: 9.0MB / W: 0.4MB	2021. 10. 7. 오후 4:58:33 (19:20:58)	i-Sujinui-MacBookPro.local	

admin

실행중 종료 기타 ... 시작									
접근키 (최대 길이 64자)									
#	User ID	세션 정보	상태	제어	구성	사용량	예약시간		
1	admin@lablup.com	AdminSession testing/repository 21.05.0 REPO	RUNNING	>_	default ^{RG} 2코어 1.00GB _GPU Demo	CPU RAM I/O R: 0.0MB / W: 0.4MB	2021. 10. 8. 오후 12:17:20 (00:00:58)		
2	user@lablup.com	UserSession testing/repository 21.05.0 REPO	RUNNING	>_	default ^{RG} 1코어 3.00GB _GPU Demo	CPU RAM I/O R: 9.0MB / W: 0.4MB	2021. 10. 7. 오후 4:58:33 (19:19:45)		

User

실행중 종료 기타 ... 시작									
#	세션 정보	상태	제어	구성	사용량	예약시간			
1	UserSession testing/repository 21.05.0 REPO	RUNNING	>_	default ^{RG} 1코어 3.00GB _GPU Demo	CPU RAM I/O R: 9.0MB / W: 0.4MB	2021. 10. 7. 오후 4:58:33 (19:21:35)			

Receiving session list

backend.ai-webui/src/components/backend-ai-session-list.ts

_viewStateChanged()



_refreshJobData()



globalThis.backendaiclient.
computeSession.list()



this.client.query()

```
async _refreshJobData(refresh = false, repeat = true) {  
  ...  
  globalThis.backendaiclient.computeSession.list(fields,  
    status, accessKey, limit, offset, group, timeout)  
  ...  
}
```

Request session list

backend.ai-webui/src/lib/backend.ai-client-node.ts

`globalThis.backendaiclient.computeSession.list()`

- query
: compute_session_list
- result
: fields 내용 (items), total_count

```
/**
 * list compute sessions with specific conditions.
 *
 * @param {array} fields - fields to query. Default fields are: ["id", "name", "image", "created_at",
"terminated_at", "status", "status_info", "occupied_slots", "cpu_used", "io_read_bytes",
"io_write_bytes"].
 * @param {string or array} status - status to query. Default is 'RUNNING'.
 * Available statuses are: `PREPARING`, `BUILDING`, `PENDING`, `SCHEDULED`, `RUNNING`,
`RESTARTING`, `RESIZING`, `SUSPENDED`, `TERMINATING`, `TERMINATED`, `ERROR`.
 * @param {string} accessKey - access key that is used to start compute sessions.
 * @param {number} limit - limit number of query items.
 * @param {number} offset - offset for item query. Useful for pagination.
 * @param {string} group - project group id to query. Default returns sessions from all groups.
 * @param {number} timeout - timeout for the request. Default uses SDK default. (5 sec.)
 */
async list(fields = ["id", "name", "image", "created_at", "terminated_at", "status", "status_info",
"occupied_slots", "containers {live_stat last_stat}"],
           status = 'RUNNING', accessKey = '', limit = 30, offset = 0, group = '', timeout: number =
0) {
  fields = this.client._updateFieldCompatibilityByVersion(fields); // For V3/V4 API compatibility
  let q, v;
  q = `query($limit:Int!, $offset:Int!, $ak:String, $group_id:String, $status:String) {
    compute_session_list(limit:$limit, offset:$offset, access_key:$ak, group_id:$group_id,
                        status:$status) {
      items { ${fields.join(" ")} }
      total_count
    }
  }`;
  v = {
    'limit': limit,
    'offset': offset,
    'status': status
  };
  if (accessKey !== '') {
    v['ak'] = accessKey;
  }
  if (group !== '') {
    v['group_id'] = group;
  }
  return this.client.query(q, v, null, timeout);
}
```


Request session list

backend.ai-webui/src/lib/backend.ai-client-node.ts

_viewStateChanged()



_refreshJobData()



globalThis.backendaiclient.
computeSession.list()



this.client.query()

```
/**
 * Send GraphQL requests
 *
 * @param {string} q - query string for GraphQL
 * @param {string} v - variable string for GraphQL
 * @param {number} timeout - Timeout to force terminate request
 * @param {number} retry - The number of retry when request is failed
 */
async query(q, v, signal = null, timeout: number = 0, retry: number = 0) {
  let query = {
    'query': q,
    'variables': v
  };
  let rqst = this.newSignedRequest('POST', `/admin/graphql`, query);
  return this._wrapWithPromise(rqst, false, signal, timeout, retry);
}
```

Request session list

backend.ai-webui/src/lib/backend.ai-client-node.ts

`this.client.query('POST', `/admin/graphql`, query)`

→ `newSignedRequest(method, queryString, body)`

- fetch() API 에 전달해주기 위해
적절한 Request 정보를 가진 object
생성

```
/**
 * Generate a RequestInfo object that can be passed to fetch() API,
 * which includes a properly signed header with the configured auth information.
 *
 * @param {string} method - the HTTP method
 * @param {string} queryString - the URI path and GET parameters
 * @param {any} body - an object that will be encoded as JSON in the request body
 */
newSignedRequest(method: string, queryString, body: any) {
  ...
  if (this._config.connectionMode === 'SESSION') {
    hdrs = new Headers({
      "User-Agent": `Backend.AI Client for Javascript
        ${this.mangleUserAgentSignature()}`,
      "X-BackendAI-Version": this._config.apiVersion,
      "X-BackendAI-Date": d.toISOString(),
    })
  } else {
    ...
    hdrs = new Headers({
      "User-Agent": `Backend.AI Client for Javascript
        ${this.mangleUserAgentSignature()}`,
      "X-BackendAI-Version": this._config.apiVersion,
      "X-BackendAI-Date": d.toISOString(),
      "Authorization": `BackendAI signMethod=HMAC-SHA256,
        credential=${this._config.accessKey}:${rqstSig}`,
    });
    uri = this._config.endpoint + queryString;
  }
  ...
  let requestInfo = {
    method: method,
    headers: hdrs,
    cache: 'default',
    body: requestBody,
    uri: uri
  };
  return requestInfo;
}
```

Request session list

backend.ai-webui/src/lib/backend.ai-client-node.ts

_viewStateChanged()



_refreshJobData()



globalThis.backendaiclient.
computeSession.list()



this.client.query()

```
/**
 * Send GraphQL requests
 *
 * @param {string} q - query string for GraphQL
 * @param {string} v - variable string for GraphQL
 * @param {number} timeout - Timeout to force terminate request
 * @param {number} retry - The number of retry when request is failed
 */
async query(q, v, signal = null, timeout: number = 0, retry: number = 0) {
  let query = {
    'query': q,
    'variables': v
  };
  let rqst = this.newSignedRequest('POST', `/admin/graphql`, query);
  return this._wrapWithPromise(rqst, false, signal, timeout, retry);
}
```

Request session list

backend.ai-webui/src/lib/backend.ai-client-node.ts

`_wrapWithPromise()`

- Backend.AI manager 로 비동기 요청을 Promise로 wrapping
- request 정보 설정 후 fetch

```
/**
 * Promise wrapper for asynchronous request to Backend.AI manager.
 *
 * @param {Request} rqst - Request object to send
 * @param {Boolean} rawFile - True if it is raw request
 * @param {AbortController.signal} signal - Request signal to abort fetch
 * @param {number} timeout - Custom timeout (sec.) If no timeout is given, default timeout
is used.
 * @param {number} retry - an integer to retry this request
 * @param {String} logText - the number of login attempts if not empty
 */
async _wrapWithPromise(rqst, rawFile = false, signal = null, timeout: number = 0,
retry: number = 0, logText = '') {
  ...
  try {
    if (rqst.method === 'GET') {
      rqst.body = undefined;
    }
    // Force request to use Public when session mode is enabled
    if (this._config.connectionMode === 'SESSION') {
      rqst.credentials = 'include';
      rqst.mode = 'cors';
    }
    if (signal !== null) {
      rqst.signal = signal;
    } else { // Use client-wide fetch timeout.
      let controller = new AbortController();
      rqst.signal = controller.signal;
      requestTimer = setTimeout(() => {
        errorType = Client.ERR_TIMEOUT;
        controller.abort();
      }, (timeout === 0 ? this.requestTimeout : timeout));
    }
    resp = await fetch(rqst.uri, rqst);
    ...
  } catch (err) {
    ...
  }
  ...
}
```


Request session list

backend.ai-webui/src/lib/backend.ai-client-node.ts

_wrapWithPromise()

- Backend.AI manager 로 비동기 요청을 Promise로 wrapping
- request 정보 설정 후 fetch
- Content Type 에 맞게 body 설정

```
if (typeof requestTimer !== "undefined") {
  clearTimeout(requestTimer);
}
errorType = Client.ERR_RESPONSE;
let contentType = resp.headers.get('Content-Type');
if (rawFile === false && contentType === null) {
  if (resp.blob === undefined)
    body = await resp.buffer(); // for node-fetch
  else
    body = await resp.blob();
} else if (rawFile === false && (contentType.startsWith('application/json') ||
  contentType.startsWith('application/problem+json'))) {
  body = await resp.json(); // Formatted error message from manager
  errorType = body.type;
  errorTitle = body.title;
} else if (rawFile === false && contentType.startsWith('text/')) {
  body = await resp.text();
} else {
  if (resp.blob === undefined) {
    body = await resp.buffer(); // for node-fetch
  } else {
    body = await resp.blob();
  }
}
errorType = Client.ERR_SERVER;
if (!resp.ok) {
  throw body;
}
```

Request session list

backend.ai-webui/src/lib/backend.ai-client-node.ts

_wrapWithPromise()

- Backend.AI manager 로 비동기 요청을 Promise로 wrapping
- request 정보 설정 후 fetch
- Content Type 에 맞게 body 설정
- log 저장 후 body 리턴

```
if(previous_log) {
  log_stack = log_stack.concat(previous_log);
}
try {
  localStorage.setItem('backendaibwebui.logs', JSON.stringify(log_stack));
} catch (e) {
  console.warn('Local storage is full. Clearing part of the logs.');
  // localStorage is full, we will keep the recent 2/3 of the logs.
  let webuiLogs = JSON.parse(localStorage.getItem('backendaibwebui.logs') || '[]');
  webuiLogs = webuiLogs.slice(0, Math.round(webuiLogs.length * 2 / 3));
  localStorage.setItem('backendaibwebui.logs', JSON.stringify(webuiLogs));
  // Deprecatd backendaiconsole.* should also be cleared here.
  Object.entries(localStorage)
    .map((x) => x[0]) // get key
    .filter((x) => x.startsWith('backendaiconsole')) // filter keys start with
    .map((x) => localStorage.removeItem(x)); // remove filtered keys
}

return body;
}
```

Receive session list

backend.ai-webui/src/components/backend-ai-session-list.ts

_refreshJobData()

- total_session_count 와 sessions 정보 저장

```
async _refreshJobData(refresh = false, repeat = true) {  
  ...  
  globalThis.backendaiclient.computeSession.list(fields, status, accessKey, limit,  
    offset, group, timeout).then((response) => {  
    this.total_session_count = response.compute_session_list.total_count;  
    ...  
  
    const sessions = response.compute_session_list.items;  
    if (sessions !== undefined && sessions.length !== 0) {  
      ...  
  
      Object.keys(sessions).map((objectKey, index) => {  
        const session = sessions[objectKey];  
        const occupied_slots = JSON.parse(session.occupied_slots);  
        const kernelImage = sessions[objectKey].image.split('/')[2] ||  
sessions[objectKey].image.split('/')[1];  
        sessions[objectKey].cpu_slot = parseInt(occupied_slots.cpu);  
        sessions[objectKey].mem_slot =  
parseFloat(globalThis.backendaiclient.utils.changeBinaryUnit(occupied_slots.mem,  
'g'));  
        sessions[objectKey].mem_slot = sessions[objectKey].mem_slot.toFixed(2);  
        ...  
      }).catch((err) => {  
        ...  
      });  
    }  
  })  
}
```

Receive session list

backend.ai-webui/src/components/backend-ai-session-list.ts

_refreshJobData()

- total_session_count 와 sessions 정보 저장
- sessions 정보 업데이트 및 새로고침 이벤트 실행 ('backend-ai-resource-refreshed')

```
async _refreshJobData(refresh = false, repeat = true) {  
  ...  
  
  globalThis.backendaiclient.computeSession.list(fields, status, accessKey,  
    limit, offset, group, timeout).then((response) => {  
    ...  
  
    this.compute_sessions = sessions;  
    this.requestUpdate();  
    let refreshTime;  
    this.refreshing = false;  
  
    if (this.active === true) {  
      if (refresh === true) {  
        const event = new CustomEvent('backend-ai-resource-refreshed',  
          {'detail': {}});  
        document.dispatchEvent(event);  
      }  
      if (repeat === true) {  
        refreshTime = this.condition === 'running' ? 5000 : 30000;  
        this.refreshTimer = setTimeout(() => {  
          this._refreshJobData();  
        }, refreshTime);  
      }  
    }  
  
    ...  
  
  }).catch((err) => {  
    ...  
  });  
}
```


Receive session list

backend.ai-webui/src/components/backend-ai-session-list.ts

render()

- this.compute_sessions에 저장된 내용을 리스트 형태로 렌더링
- admin 계정이면 access_key 혹은 user_email 도 보여줌
- superadmin 계정이면 agent 정보도 보여줌

```
<vaadin-grid id="list-grid" theme="row-stripes column-borders compact"
  aria-label="Session list" .items="${this.compute_sessions}" height-by-rows>
  ...
  ${this.is_admin ? html`
    <vaadin-grid-filter-column path="${this._connectionMode === 'API' ? 'access_key' : 'user_email'}"
      header="${this._connectionMode === 'API' ? 'API Key' : 'User ID'}"
      .renderer="${this._boundUserInfoRenderer}" resizable>
    </vaadin-grid-filter-column>
  ` : html``}
  ...
  ${this.is_superuser ? html`
    <vaadin-grid-column auto-width flex-grow="0" resizable header="${_t('session.Agent')}"
      .renderer="${this._boundAgentRenderer}">
    </vaadin-grid-column>
  ` : html``}
</vaadin-grid>
```

Superadmin

실행중 종료 기타 ... 시작								
접근키 (최대 길이 64자)								
#	User ID	세션 정보	상태	제어	구성	사용량	예약시간	실행노드
1	admin@lablup.com	AdminSession testing/repository REPO	RUNNING		default ^{RG} 2코어 1.00GB _GPU Demo	CPU RAM I/O R: 0.0MB / W: 0.4MB	2021. 10. 8. 오후 12:17:20 (00:02:11)	i-Sujinui-MacBookPro.local
2	user@lablup.com	UserSession testing/repository REPO	RUNNING		default ^{RG} 1코어 3.00GB _GPU Demo	CPU RAM I/O R: 9.0MB / W: 0.4MB	2021. 10. 7. 오후 4:58:33 (19:20:58)	i-Sujinui-MacBookPro.local

admin

실행중 종료 기타 ... 시작								
접근키 (최대 길이 64자)								
#	User ID	세션 정보	상태	제어	구성	사용량	예약시간	
1	admin@lablup.com	AdminSession testing/repository 21.05.0 REPO	RUNNING		default ^{RG} 2코어 1.00GB _GPU Demo	CPU RAM I/O R: 0.0MB / W: 0.4MB	2021. 10. 8. 오후 12:17:20 (00:00:58)	
2	user@lablup.com	UserSession testing/repository 21.05.0 REPO	RUNNING		default ^{RG} 1코어 3.00GB _GPU Demo	CPU RAM I/O R: 9.0MB / W: 0.4MB	2021. 10. 7. 오후 4:58:33 (19:19:45)	

User

실행중 종료 기타 ... 시작							
#	세션 정보	상태	제어	구성	사용량	예약시간	
1	UserSession testing/repository 21.05.0 REPO	RUNNING		default ^{RG} 1코어 3.00GB _GPU Demo	CPU RAM I/O R: 9.0MB / W: 0.4MB	2021. 10. 7. 오후 4:58:33 (19:21:35)	

“ Thank you (´•ᵛ•`) ”